AIR NAVIGATION/*RNAV*
NAVIGATION FORMULAS

DISTANCE: The Great Circle distance between fixes is calculated by the following formulas:

$$d=COS^{-1}[SIN(LAT)SIN(LAT')+COS(LAT)COS(LAT')COS(LNG-LNG')]$$

$$(Distance\ in\ NM)=60d$$

Where: LAT' and LNG' are the coordinates of the "from" fix.
LAT and LNG are the coordinates of the "to" fix.

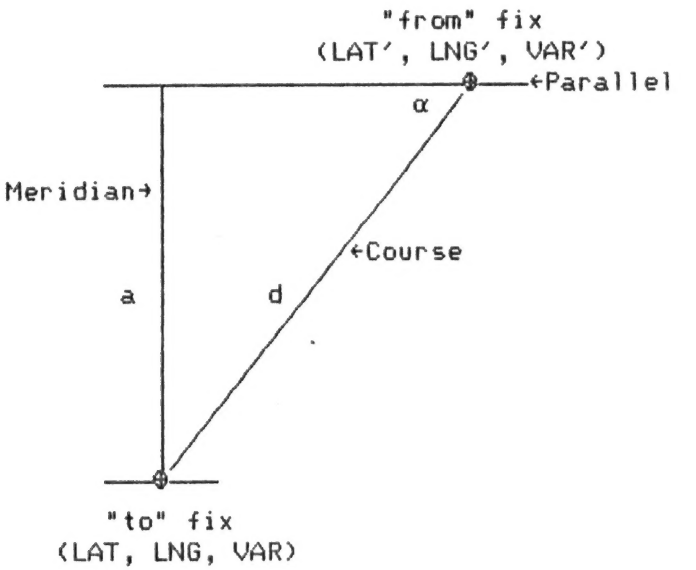
Using Atari functions the formula for d is:

$$d=90-TAN^{-1}[X/(1-X^2)^2]$$
$$X=[SIN(LAT)SIN(LAT')+COS(LAT)COS(LAT')COS(LNG-LNG')]$$

AIR NAVIGATION/*RNAV*
NAVIGATION FORMULAS

COURSE: The course is derived by finding the angle "α" between the parallel passing through the "from" fix and the Course. The angle "α" is then corrected for East/West direction of travel.

A diagram illustrating the relationships is given below with the formulas:



$a=COS^{-1}(Y)$ Atari functions: $a=90-TAN^{-1}[Y/(1-Y^2)^2]$
Where $Y=[SIN(LAT)SIN(LAT')+COS(LAT)COS(LAT')]$

$d=COS^{-1}(X)$ Atari functions: $d=90-TAN^{-1}[X/(1-X^2)^2]$
Where $X=[SIN(LAT)SIN(LAT')+COS(LAT)COS(LAT')COS(LNG-LNG')]$

$\alpha=SIN^{-1}(A)$ Atari functions: $\alpha=TAN^{-1}[A/(1-A^2)^2][SGN(LAT-LAT')]$
Where $A=SIN(a)/SIN(d)$

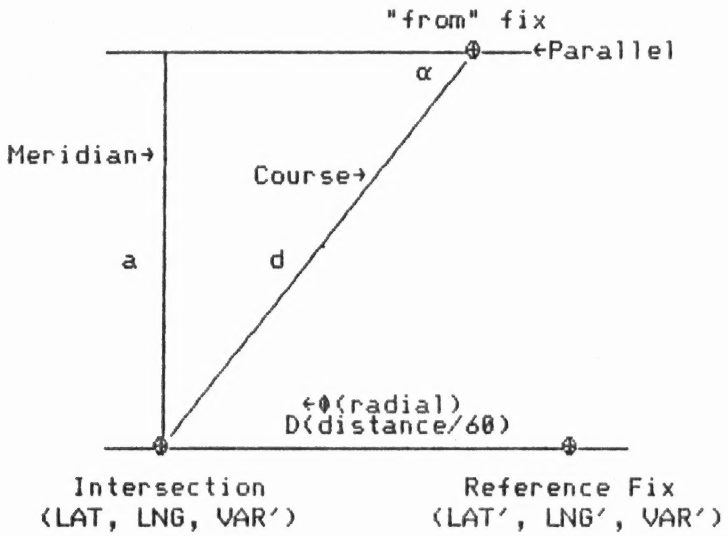
If $(LNG'-LNG)<0$ then $TC=270+\alpha$ (Westerly direction of travel.)
If $(LNG'-LNG)\geq 0$ then $TC=90-\alpha$ (Easterly direction of travel.)

MC=TC+VAR′, and if VAR*VAR′ then MC=TC+VAR′/TC+VAR

AIR NAVIGATION/*RNAV*
NAVIGATION FORMULAS

INTERSECTIONS: Intersections are always defined by the radial (magnetic bearing) and distance from a known fix (reference navaid). The program uses this information along with the coordinates of the reference fix to derive the coordinates of the intersection. These coordinates are then used the same as the coordinates of any other fix.

A diagram illustrating the relationships and the formulas is given below:



$LAT=LAT'+SIN^{-1}[(SIN\ D)(COS\ \Phi)]$

$LNG=LNG'-SIN^{-1}[[(SIN\ D)(SIN\ \Phi)]/[COS[(LAT+LAT')/2]]$

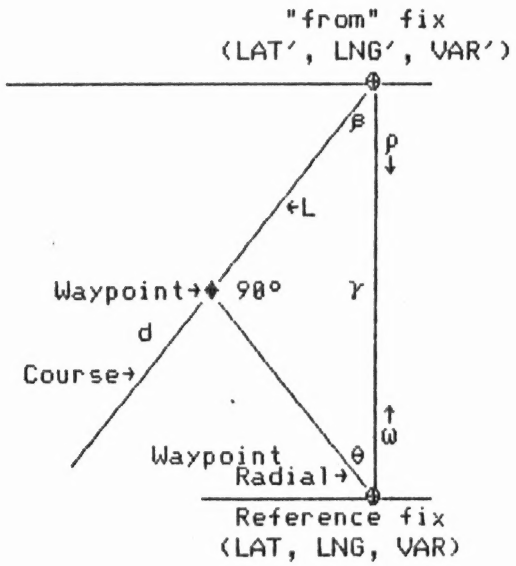
The formulas using Atari functions:

$LAT=LAT'+TAN^{-1}[X/(1-X^2)^2]$
Where $X=(SIN\ D)(COS\ \Phi)$

$LNG=LNG'-TAN^{-1}[Y/(1-Y^2)^2]$
Where $Y=[(SIN\ D)(SIN\ \Phi)]/COS[(LAT+LAT')/2]$

AIR NAVIGATION/*RNAV*
NAVIGATION FORMULAS

WAYPOINTS: Waypoints are calculated by deriving the bearing and distance of the waypoint from the reference navaid along the radial that is perpendicular to the course. This will form a spherical right triangle between the waypoint, reference navaid, and the "from" fix:



The bearing (ρ) and distance angle (γ) from the "from" fix to the reference fix is calculated first. The angle, γ, is calculated with the same formula as used to find the course distance angle (d), and ρ is calculated the same way as the true course. The bearing (ω) from the reference fix to the "from" fix and is the reciprocal of ρ, $\omega=\rho\pm180$. The angle, β, is the difference between the bearing (ρ) and the true course. This angle is expressed as:

$$-180° \leq \beta \leq 180°$$

If $\beta > |90°|$ then the way point will not be usable because it will occur the other side of the "from" fix.

AIR NAVIGATION/*RNAV*
NAVIGATION FORMULAS

The distance (L) between the "from" fix and the waypoint is calculated with the following formula:

$$L=60[SIN^{-1}[SIN(|\theta|)SIN(\gamma)]] \qquad \theta=90-\beta$$

Using the Atari functions the formula becomes:

$$L=60[TAN^{-1}[X/(1-X^2)^2]]$$

Where $X=SIN(|\theta|)SIN(\gamma)$

The waypoint radial and distance are calculated with the following formulas:

$$\text{Waypoint radial} = \omega+SGN(\beta)(\theta)+VAR$$

$$\text{Waypoint distance} = 60[SIN^{-1}[SIN(|\beta|)SIN(\gamma)]$$

Using Atari functions:

$$\text{Waypoint distance} = 60[TAN^{-1}[Y/(1-Y^2)^2]$$

Where $Y=SIN(|\beta|)SIN(\gamma)]$

TERMINAL WAYPOINTS

The terminal waypoints are calculated by letting the reference navaid be the "from" fix and the terminal airport be the "to" fix. The same formulas used to find the true course and distance are used. The reference navaid's magnetic variation is added to the bearing from the reference fix to the terminal airport to obtain the proper radial. The reference navaid's magnetic variation is also used to obtain the magnetic course from the departure airport or to the destination airport as the case may be.

W I N D C O R R E C T I O N H P - 4 1 C P R O G R A M

Note: All velocities are absolute values.

True Heading:

$$TH=TC+SIN^{-1}[Wv*SIN(Wd-TC)/TAS]$$

Wv=Wind Velocity
Wd=Wind Direction

Ground Speed:

$$GS=[(TAS*SIN(TH)-Wv*SIN(Wd))^2+(TAS*COS(TH)-Wv*COS(Wd))^2]^{½}$$

If COS(Wd-TH) is negative and TAS<Wv then the GS is negative.

Wind Direction:

$$Wd=TAN^{-1}[(TAS*SIN(TH)-GS*SIN(TC))/(TAS*COS(TH)-GS*COS(TC))]$$

If TAS*COS(TH)-GS*COS(TC) is negative then add 180 to the Wd.

Wind Velocity:

$$Wv=[(TAS*SIN(TH)-GS*SIN(TC))^2+(TAS*COS(TH)-GS*COS(TC))^2]^{½}$$

Data Registers:

R00 - Time in HR format
R01 - Wd
R02 - Wv
R03 - TC
R04 - TAS
R05 - TH
R06 - GS
R07 - Distance

AIR NAVIGATION/*RNAV*
NAVIGATION FORMULAS

DISTANCE: The Great Circle distance between fixes is calculated by the following formulas:

$$d=\cos^{-1}[\sin(LAT)\sin(LAT')+\cos(LAT)\cos(LAT')\cos(LNG-LNG')]$$

$$(\text{Distance in NM})=60d$$

Where: LAT′ and LNG′ are the coordinates of the "from" fix.
LAT and LNG are the coordinates of the "to" fix.

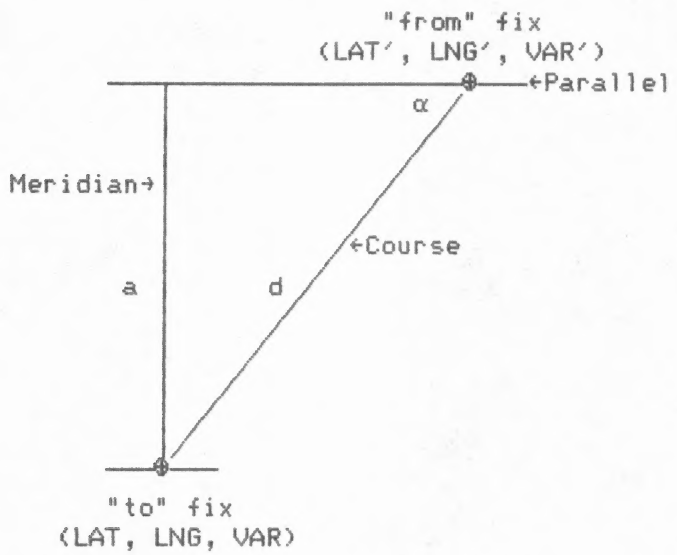
Using Atari functions the formula for d is:

$$d=90-\tan^{-1}[X/(1-X^2)^2]$$
$$X=[\sin(LAT)\sin(LAT')+\cos(LAT)\cos(LAT')\cos(LNG-LNG')]$$

AIR NAVIGATION/*RNAV*
NAVIGATION FORMULAS

COURSE: The course is derived by finding the angle "α" between the parallel passing through the "from" fix and the Course. The angle "α" is then corrected for East/West direction of travel.

A diagram illustrating the relationships is given below with the formulas:



$a=COS^{-1}(Y)$ Atari functions: $a=90-TAN^{-1}[Y/(1-Y^2)^2]$
Where $Y=[SIN(LAT)SIN(LAT')+COS(LAT)COS(LAT')]$

$d=COS^{-1}(X)$ Atari functions: $d=90-TAN^{-1}[X/(1-X^2)^2]$
Where $X=[SIN(LAT)SIN(LAT')+COS(LAT)COS(LAT')COS(LNG-LNG')]$

$\alpha=SIN^{-1}(A)$ Atari functions: $\alpha=TAN^{-1}[A/(1-A^2)^2][SGN(LAT-LAT')]$
Where $A=SIN(a)/SIN(d)$

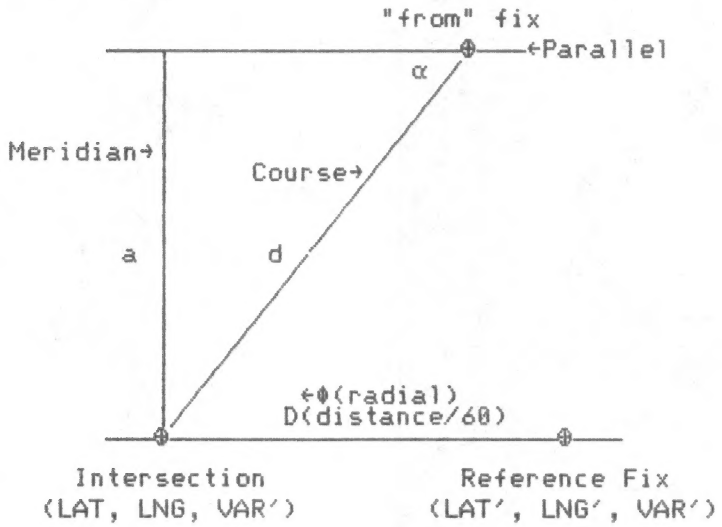
If $(LNG'-LNG)<0$ then $TC=270+\alpha$ (Westerly direction of travel.)
If $(LNG'-LNG)\geq 0$ then $TC=90-\alpha$ (Easterly direction of travel.)

MC=TC+VAR', and if VAR≠VAR' then MC=TC+VAR'/TC+VAR

AIR NAVIGATION/*RNAV*
NAVIGATION FORMULAS

INTERSECTIONS: Intersections are always defined by the radial (magnetic bearing) and distance from a known fix (reference navaid). The program uses this information along with the coordinates of the reference fix to derive the coordinates of the intersection. These coordinates are then used the same as the coordinates of any other fix.

A diagram illustrating the relationships and the formulas is given below:



$LAT=LAT'+SIN^{-1}[(SIN\ D)(COS\ \Phi)]$

$LNG=LNG'-SIN^{-1}[[(SIN\ D)(SIN\ \Phi)]/[COS[(LAT+LAT')/2]]$

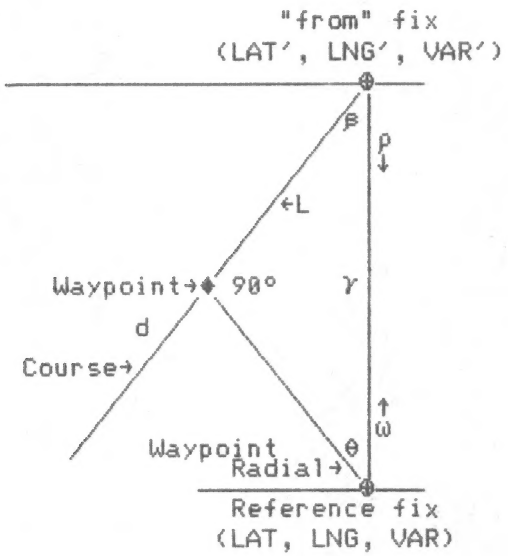
The formulas using Atari functions:

$LAT=LAT'+TAN^{-1}[X/(1-X^2)^2]$
Where $X=(SIN\ D)(COS\ \Phi)$

$LNG=LNG'-TAN^{-1}[Y/(1-Y^2)^2]$
Where $Y=[(SIN\ D)(SIN\ \Phi)]/COS[(LAT+LAT')/2]$

AIR NAVIGATION/*RNAV*
NAVIGATION FORMULAS

WAYPOINTS: Waypoints are calculated by deriving the bearing and distance of the waypoint from the reference navaid along the radial that is perpendicular to the course. This will form a spherical right triangle between the waypoint, reference navaid, and the "from" fix:



The bearing (ρ) and distance angle (γ) from the "from" fix to the reference fix is calculated first. The angle, γ, is calculated with the same formula as used to find the course distance angle (d), and ρ is calculated the same way as the true course. The bearing (ω) from the reference fix to the "from" fix and is the reciprocal of ρ, ω=ρ±180. The angle, β, is the difference between the bearing (ρ) and the true course. This angle is expressed as:

$$-180° \leq \beta \leq 180°$$

If $\beta > |90°|$ then the way point will not be usable because it will occur the other side of the "from" fix.

NAVIGATION FORMULAS

The distance (L) between the "from" fix and the waypoint is calculated with the following formula:

$$L=60[SIN^{-1}[SIN(|\theta|)SIN(\gamma)]] \qquad \theta=90-\beta$$

Using the Atari functions the formula becomes:

$$L=60[TAN^{-1}[X/(1-X^2)^2]]$$

Where $X=SIN(|\theta|)SIN(\gamma)$

The waypoint radial and distance are calculated with the following formulas:

$$\text{Waypoint radial} = \omega+SGN(\beta)(\theta)+VAR$$

$$\text{Waypoint distance} = 60[SIN^{-1}[SIN(|\beta|)SIN(\gamma)]$$

Using Atari functions:

$$\text{Waypoint distance} = 60[TAN^{-1}[Y/(1-Y^2)^2]$$

Where $Y=SIN(|\beta|)SIN(\gamma)]$

TERMINAL WAYPOINTS

The terminal waypoints are calculated by letting the reference navaid be the "from" fix and the terminal airport be the "to" fix. The same formulas used to find the true course and distance are used. The reference navaid's magnetic variation is added to the bearing from the reference fix to the terminal airport to obtain the proper radial. The reference navaid's magnetic variation is also used to obtain the magnetic course from the departure airport or to the destination airport as the case may be.

```
100 POKE 710,0:DIM FROM$(10),DEST$(10)
110 ? "}":? "WHAT IS YOUR DEPARTURE POINT";:INPUT FROM$:IF LEN(FROM$)=0 THEN 110
120 TRAP 110:? "WHAT IS THE ";FROM$;" LATITUDE";:INPUT X:GOSUB 200:LAT1=X
130 ? "WHAT IS THE ";FROM$;" LONGITUDE";:INPUT X:GOSUB 200:LNG1=X
150 ? "}":? "WHAT IS YOUR DESTINATION POINT";:INPUT DEST$:IF LEN(DEST$)=0 THEN 150
160 TRAP 150:? "WHAT IS THE ";DEST$;" LATITUDE";:INPUT X:GOSUB 200:LAT2=X
170 ? "WHAT IS THE ";DEST$;" LONGITUDE";:INPUT X:GOSUB 200:LNG2=X
190 TRAP 40000:DEG :GOTO 240:REM CLEAR ERROR TRAP, SET DEGREE MODE
200 REM CONVERT MINUTES AND SECONDS TO DECIMAL
210 Y=INT(X):X=100*(X-Y)
220 X=X/60:X=Y+X
230 RETURN
240 POKE 752,1:? "}":? "I AM CALCULATING!":? :? "          -THEO":POKE 752,0
250 X=SIN(LAT1)*SIN(LAT2)+COS(LAT1)*COS(LAT2)*COS(LNG2-LNG1)
260 REM ARCCOSX=90-ARCTAN[X/SQR((-X*X)+1]
270 DISTANGLE=90-ATN(X/SQR(-X*X+1))
275 D=DISTANGLE*60
280 X=(SIN(LAT2)-SIN(LAT1)*COS(D/60))/(SIN(D/60)*COS(LAT1))
290 TC=90-ATN(X/SQR(-X*X+1))
300 IF SIN(LNG1-LNG2)<0 THEN TC=360-TC
360 ? "}":? "FROM ";FROM$;" TO ";DEST$;":"
370 REM ROUND OFF AND PRINT RESULTS
380 ? "THE DISTANCE IS ";INT(60*DISTANGLE+0.5);" NM."
390 ? "THE TRUE COURSE IS ";INT(TC+0.5);" Degrees"
```

AIR NAVIGATION/*RNAV*
PRINTER CODES

1 Printer Initialization:

027 069 027 076 048 048 056 027 065 027 060 027

034 046 000 000 000 000 000 000 000 000 000 000

2 Degree Symbol:

027 083 048 048 056 000 006 009 009 006 000 000

000 000 000 000 000 000 000 000 000 000 000 000

3 Single Line (±): 149 Double Line (½): 224

Enter Field to Change - or RETURN to Accept

ESC to abort

```
DEFINT A-Z
SUB Facility STATIC
   Fac$ = "NAVAID ": IF Id THEN Fac$ = "AIRPORT": IF Id < 1 THEN Fac$ = "DELETED"
   IF Id > 1 THEN Fac$ = "WAYPOINT": IF Id = 3 THEN Fac$ = "INTERSECTION"
'NOTES:
   '0 = NAVAID
   '1 = AIRPORT
   '2 = WAYPOINT
   '3 = INTERSECTION
   '4 = TERMINAL WAYPOINT
   '5 = END OF FLIGHT
END SUB

DEFINT A-Z
FUNCTION Increment (Index, CoordinateFlag) STATIC
   Increment = Index + 6 + CoordinateFlag * 27 + 15 * -(Id = 3)
'NOTES: Flight$ (Flight route data)
   'Navaid, Airport, or Waypoint -
      'Byte 1:                    Fix identifier (see Sub Facility)
      'Bytes 2 to 5:              Four letter identifier
      'Byte 6:                    Coordinate Flag (0 = in NavFile, 1 = in Flight$)
      'Bytes 7 to 33:             Coordinate data if stored in the Flight$
      '  Bytes 7 to 14:              Latitude (8 bytes - bytes 1 to 8 relative)
      '  Bytes 15 to 23:             Longtitude (9 bytes - bytes 9 to 17 relative)
      '  Bytes 24 to 26:             Variation (3 bytes - bytes 18 to 20 relative)
      '  Bytes 27 to 33:             Frequence (7 bytes - bytes 21 to 27 relative)
   'Intersection -
      'Byte 1:                    "3" for intersection
      'Bytes 2 to 6:              Five letter identifier
      'Bytes 7 to 11(38):         Reference Navaid Data
      '  Bytes 7 to 10:              Four letter identifier
      '  Byte 11:                    Coordinate Flag
      '  Bytes (12) to (38):         Coordinate data if stored in the Flight$
      'Bytes 12(39) to 16(43):    Radial (5 bytes - 000.1 to 360.0)
      'Bytes 17(44) to 21(48):    DME Distance (5 bytes - 0.0 to 999.9)
END FUNCTION
```

RICHARD TUTTLE
NATIONWIDE INSURANCE
541 KIBLER AVE.
NEWARK, OHIO 43055

AIR NAVIGATION PROGRAM

```
10 REM AIR NAVIGATION PROGRAM - TRUE COURSE & DISTANCE
20 REM by ROBERT A. TIMS, JONESBORO, AR
30 REM POKE COMMANDS ARE FOR ATARI DISPLAY ONLY, AND MAY BE IGNORED.
40 REM FROM FIX IS LAT1,LNG1: DEST FIX is LAT2,LNG2
50 REM DISTANCE ANGLE=ArcCos[SinLAT1*SinLAT2+CosLAT1*CosLAT2*Cos(LNG2-LNG1)]
60 REM DISTANCE=60*DISTANGLE
70 REM LNG ANGLE=ArcCos(SinLat1*SinLat2+CosLat1*CosLat2)
80 REM ANGLE=ArcSin(SinLNGANGLE/SinDISTANGLE)
90 REM TC=270+ANGLE or 90-ANGLE
100 POKE 710,0:DIM FROM$(10),DEST$(10)
110 ? "}":? "WHAT IS YOUR DEPARTURE POINT";:INPUT FROM$:IF LEN(FROM$)=0 THEN 110
120 TRAP 110:? "WHAT IS THE ";FROM$;" LATITUDE";:INPUT X:GOSUB 200:LAT1=X
130 ? "WHAT IS THE ";FROM$;" LONGITUDE";:INPUT X:GOSUB 200:LNG1=X
150 ? "}":? "WHAT IS YOUR DESTINATION POINT";:INPUT DEST$:IF LEN(DEST$)=0 THEN 150
160 TRAP 150:? "WHAT IS THE ";DEST$;" LATITUDE";:INPUT X:GOSUB 200:LAT2=X
170 ? "WHAT IS THE ";DEST$;" LONGITUDE";:INPUT X:GOSUB 200:LNG2=X
190 TRAP 40000:DEG :GOTO 240:REM CLEAR ERROR TRAP, SET DEGREE MODE
200 REM CONVERT MINUTES AND SECONDS TO DECIMAL
210 Y=INT(X):X=100*(X-Y)
220 Y=Y+INT(X)/60:X=100*(X-INT(X))
230 X=INT(X)/3600+Y:RETURN
240 POKE 752,1:? "}":? "I AM CALCULATING!":? :? "            - THEO":POKE 752,0
250 X=SIN(LAT1)*SIN(LAT2)+COS(LAT1)*COS(LAT2)*COS(LNG2-LNG1)
260 REM ArcCosX=90-ArcTan[X/SQR((-X*X)+1)]
270 DISTANGLE=90-ATN(X/SQR(-X*X+1))
280 X=SIN(LAT1)*SIN(LAT2)+COS(LAT1)*COS(LAT2)
290 IF X)=1 THEN LNGANGLE=0:GOTO 310
300 LNGANGLE=90-ATN(X/SQR(-X*X+1))
310 X=SIN(LNGANGLE)/SIN(DISTANGLE):IF X=1 THEN ANGLE=SGN(LAT2-LAT1)*90:GOTO 340
320 REM ArcSinX=ArcTan[X/SQR((-X*X)+1)]
330 ANGLE=ATN(X/SQR(-X*X+1))*SGN(LAT2-LAT1)
340 IF SGN(LNG2-LNG1)(0 THEN TC=90-ANGLE:GOTO 360
350 TC=270+ANGLE
360 ? "}":? "FROM ";FROM$;" TO ";DEST$;":"
370 REM ROUND OFF AND PRINT RESULTS
380 ? "THE DISTANCE IS ";INT(60*DISTANGLE+0.5);" NM."
390 ? "THE TRUE COURSE IS ";INT(TC+0.5);" Degrees"
400 REM ****IMPROVEMENTS****
410 REM USE LINES 140 AND 180 TO ENTER MAGNETIC VARIATION AND ADD MC OUTPUT
420 REM CREATE A FILE OF NAVIGATION FIXES TO BE ACCESSED BY THE PROGRAM
430 REM SEARCH THE FILE FOR LAT,LNG,& VAR USING THE THREE LETTER DESIGNATION
500 REM ****ADDITIONAL PROJECTS****
510 REM INTERPOLATE FOR WIND AT ALTITUDE
520 REM WIND IS GIVEN AT THE FOLLOWING ALTITUDES IN THOUSANDS OF FEET:
530 REM 3,6,9,12,18,24,30,34,39,45,53: USE 3 THROUGH 18
540 REM FIND WCA: =ArcSin[Wind Velocity*Sin(Wind Direction/TAS)]
550 REM FIND MH: =MC+WCA
560 REM FIND GS: =TAS*Cos(WCA)-Wind Velocity*Cos(Wind Direction-TC)
570 REM TIME DISPLAY IN HRS:MINUTES
580 REM DEGREE DISPLAY FOR 360 TO 360 DEGREES ONLY. note: 0 degress = 360
590 REM PROGRAM INCORPORATING ALL THE ABOVE FOR A COMPLETE TRIP
```

```
'AIR NAVIGATION/*RNAV* - QuickBasic Source Code - Copyright 1989 ROBERT A. TIMS & ABS Logic, Inc.
'1-27-89 Version Date - Program File Name: AIRNAV.BAS

'Subprogram and Function Declarations
   DECLARE FUNCTION ACos# (X#)
   DECLARE FUNCTION ASin# (X#)
   DECLARE FUNCTION FormatId$ (A$)
   DECLARE FUNCTION Increment% (A%, CoordinateFlag%)
   DECLARE FUNCTION Zero$ (A$)
   DECLARE SUB AiportNavaid ()
   DECLARE SUB Continue (Row%)
   DECLARE SUB Escape (Msg$)
   DECLARE SUB Facility ()
   DECLARE SUB FieldChoice (UpperLimit%, Row%, PageFlag%)
   DECLARE SUB FileCheck ()
   DECLARE SUB FlightDisplay ()
   DECLARE SUB FlightIndex ()
   DECLARE SUB FormatDisplay ()
   DECLARE SUB IndexFile ()
   DECLARE SUB IntroScreen ()
   DECLARE SUB LatFormat ()
   DECLARE SUB LatLngFormat ()
   DECLARE SUB LngFormat ()
   DECLARE SUB Menu (UpperLimit%)
   DECLARE SUB Navaid ()
   DECLARE SUB NavFile ()
   DECLARE SUB NavRec ()
   DECLARE SUB Query (Row%, Col%, DataType%, Length%)
   DECLARE SUB RouteScreen (WaypointFlag%)
   DECLARE SUB ScreenHeading ()
   DECLARE SUB Strip ()
   DECLARE SUB SysFile ()
   DECLARE SUB TempNavFile ()
   DECLARE SUB Time (Seconds!)
   DECLARE SUB VarFormat ()
   DECLARE SUB Wipe (Row%, Col%)

'Global and Constant Data
   DEFINT A-Z
   COMMON SHARED AF$, Cr$, Dat$, Deg$, Degree$, Departure$, Destination$, Esc$, Fac$, Flight$, Format$, Freq$
   COMMON SHARED Heading$, Id$, Id1$, Id2$, Index$, K$, Lat$, Lng$, Ndx$, Old$, Rec$, Rtn$, Text$, Var$
   COMMON SHARED Edit, EscCol, Format, Hue, Id, Index, K, Max, Ndx, NRec, Rec, Direction
   DIM SHARED C(6), CI(6): WIDTH "LPT1:", 255
   CONST PI# = 3.141592653589793#
   CONST D90# = PI# / 2 'RADIANS IN A RIGHT ANGLE
   CONST D360# = 2 * PI# 'RADIANS FOR A CIRCLE
   CONST D2R# = PI# / 180 'CONSTANT TO CONVERT DEGREES TO RADIANS
   CONST R2D# = 180 / PI# 'CONSTANT TO CONVERT RADIANS TO DEGREES
   CONST RS# = R2D# * 60 'CONSTANT TO CONVERT EARTH DEGREES TO NM
   CONST No = 0, Yes = NOT No
   DegreeSymbol$ = CHR$(248): Degree$ = DegreeSymbol$: Esc$ = CHR$(27): Rtn$ = CHR$(13)

'System Parameters
   ON ERROR GOTO 0: SysFile
   CFlg = VAL(CFlg$): PrInit$ = Pr1$: PrDeg$ = Pr2$: Format = VAL(Format$)
   SingleLine$ = SL$: DoubleLine$ = DL$: CLOSE
   C$ = "777777": IF CFlg THEN C$ = "465312"
   FOR I = 0 TO 5: C(I) = VAL(MID$(C$, I + 1, 1)): CI(I) = C(I) + 8: NEXT
   C(6) = 7: CI(6) = 7: IF INKEY$ = "" THEN IntroScreen
   IF CFlg = 0 THEN FOR I = 0 TO 5: CI(I) = C(I): NEXT

'Index
   ON ERROR GOTO IndexFileError: OPEN "NavFile.NDX" FOR INPUT AS 2
   INPUT #2, Index$: CLOSE : ON ERROR GOTO 0
   CALL NavFile: IF LOF(1) = 0 THEN Text$ = "  GO TO MAIN MENU": LatLngFormat

MainMenu:
   CLOSE : Heading$ = "MAIN MENU": ScreenHeading
   LOCATE 8, 32: COLOR 0, C(1): PRINT " 1 "; : COLOR CI(1), 0: PRINT "  ROUTE ENTRY"
   LOCATE 10, 32: COLOR 0, C(2): PRINT " 2 "; : COLOR CI(2), 0: PRINT "  LOAD FLIGHT"
   LOCATE 12, 32: COLOR 0, C(3): PRINT " 3 "; : COLOR CI(3), 0: PRINT "  AIRNAV DATA"
   LOCATE 15, 32: COLOR 0, 7: PRINT " 4 "; : COLOR 7, 0: PRINT "  Printer Codes"
   LOCATE 17, 32: COLOR 0, 7: PRINT " 5 "; : COLOR 7, 0: PRINT "  Screen Setup"
   LOCATE 20, 32: COLOR 0, C(0): PRINT " 0 "; : COLOR CI(0), 0: PRINT "  QUIT"
   Menu 5: ON K GOTO RouteEntry, FlightLoad, NavDataEntry, PrinterCodes, ScreenInit
   IF K$ = "0" THEN COLOR 7, 0: CLS : END ELSE GOTO MainMenu
```

```
NavFix:
   IF Ndx THEN NavRec: Fix$ = Fix$ + Id$ + "0": RETURN
   Rec = NRec: Wipe 5, 1: FormatDisplay: DataAdded = No: GOSUB AddFacData: IF Max = 1 THEN RETURN
   IF Rec < 1000 THEN
      Wipe 21, 1: LOCATE 5, 22: COLOR CI(Hue): PRINT "FILE": LOCATE 21, 26
      PRINT "File with the Nav Data (Y/N)? "; : COLOR CI(Hue) + 16: PRINT CHR$(22): Escape "entry"
      DO
         K$ = UCASE$(INPUT$(1)): IF K$ = Esc$ THEN Max = 1: RETURN
      LOOP WHILE K$ <> "Y" AND K$ <> "N"
      IF K$ = "Y" THEN Fix$ = Fix$ + Id$ + "0": GOTO StoreNavdata
   END IF: Fix$ = Fix$ + Id$ + "1" + Lat$ + Lng$ + Var$ + Freq$: RETURN

GetIntersection:
   DO
      Temp$ = Fix$ + LEFT$(LastFix$ + "     ", 5): Msg$ = LastFix$ + " REFERENCE NAVAID"
         GOSUB DisplayFacility: Id = 0: Facility: LSET AF$ = "0"
      Edit = No: GOSUB Identifier: IF Max = 1 THEN RETURN
      Fix$ = "": GOSUB NavFix: IF K$ = Esc$ THEN RETURN
   LOOP WHILE Max = 1: Radial$ = "": Distance$ = ""
   Msg$ = LastFix$ + " INTERSECTION": GOSUB DisplayFacility: LOCATE 9, 30: PRINT "Reference Navaid:"; Id$
   FOR I = 1 TO 2
      ON I GOSUB GetRadial, GetDistance: IF Max = 1 THEN GOTO GetIntersection
   NEXT
   DO
      FieldChoice 2, 21, 0: IF K$ = Esc$ THEN Max = 1: RETURN
      IF K$ = Rtn$ THEN Fix$ = Temp$ + Fix$ + Radial$ + Distance$: RETURN
      Wipe 21, 1: Edit = Yes: ON VAL(K$) GOSUB EditRadial, EditDistance
   LOOP

GetRadial:
   LOCATE 14, 30: COLOR 0, C(Hue): PRINT " 1 "; : COLOR CI(Hue), 0: PRINT "  Radial:"
EditRadial:
   IF VAL(Radial$) = 0 THEN Text$ = "" ELSE Text$ = STR$(VAL(Radial$)): Strip
   Wipe 14, 43: Query 14, 43, 15, 5: IF Max = 1 THEN IF NOT Edit THEN RETURN ELSE Text$ = Radial$
   T! = VAL(Text$): T! = (T! * 10 MOD 3600) / 10: IF T! = 0 THEN T! = 360
   Text$ = STR$(T! + 1000): Text$ = RIGHT$(Text$, LEN(Text$) - 2): IF T! = INT(T!) THEN Text$ = Text$ + ".0"
   Radial$ = LEFT$(Text$ + "   ", 5): LOCATE 14, 43: PRINT Text$ + DegreeSymbol$: RETURN

GetDistance:
   LOCATE 16, 30: COLOR 0, C(Hue): PRINT " 2 "; : COLOR CI(Hue), 0: : PRINT "  Distance:"
EditDistance:
   IF VAL(Distance$) = 0 THEN Text$ = "" ELSE Text$ = STR$(VAL(Distance$)): Strip
   Wipe 16, 45: Query 16, 45, 15, 5: IF Max = 1 THEN IF NOT Edit THEN RETURN ELSE Text$ = Distance$
   T! = VAL(Text$): Text$ = STR$(T!): Text$ = RIGHT$(Text$, LEN(Text$) - 1)
   IF T! = INT(T!) THEN Text$ = Text$ + ".0"
   Distance$ = LEFT$(Text$ + "   ", 5): LOCATE 16, 45: PRINT Text$ + " NM": RETURN

TerminalWaypoint:
   IF Id = 1 THEN
      DO
         Msg$ = LastFix$ + " TERMINAL WAYPOINT": GOSUB DisplayFacility
         Id = 0: LSET AF$ = "0": Fix$ = "": Edit = No: GOSUB Identifier: IF Max = 1 THEN RETURN
         CALL Facility: GOSUB NavFix
      LOOP WHILE Max = 1: Flight$ = Flight$ + "4" + Fix$
   END IF: RETURN

DisplayFacility:
   CALL ScreenHeading: LOCATE 5, (80 - LEN(Msg$)) / 2: PRINT Msg$: Escape "entry": RETURN

RouteEntry:
   Editing = No: Routing = Yes: NavFile
Departure:
   Departure = Yes: OKtoEnd = No
   DO
      Hue = 1: Heading$ = "ROUTE PLANNING": ScreenHeading: RouteScreen 0
      IF K$ = Esc$ OR K$ = Rtn$ THEN GOTO MainMenu
      CALL Facility: Msg$ = "DEPARTURE " + UCASE$(Fac$): GOSUB DisplayFacility
      Fix$ = RIGHT$(STR$(Id), 1): LSET AF$ = Fix$: Edit = No: GOSUB Identifier
   LOOP WHILE Max = 1
   IF Id < 2 THEN GOSUB NavFix: LastFix$ = FormatId(Id$) ELSE LastFix$ = Text$: GOSUB GetIntersection
   IF Max = 1 THEN GOTO Departure ELSE Flight$ = Fix$: GOSUB TerminalWaypoint
Enroute:
   Departure = No: Msg$ = "LAST FIX WAS " + LastFix$: GOSUB DisplayFacility: RouteScreen 1
   IF K$ = Rtn$ AND OKtoEnd THEN
      GOSUB TerminalWaypoint: Flight$ = Flight$ + "5": GOSUB SortNavData: FlightDisplay
```

```
      IF K$ = Esc$ THEN GOTO MainMenu ELSE IF K$ = "N" THEN GOTO RouteEntry
   ELSE
      IF K$ = Rtn$ OR K$ = Esc$ THEN GOSUB SortNavData: GOTO MainMenu
      Fix$ = RIGHT$(STR$(Id), 1): Facility: Msg$ = "ENROUTE " + UCASE$(Fac$): GOSUB DisplayFacility
      LSET AF$ = Fix$: IF K = 4 THEN Id = 0: LSET AF$ = "0": Facility
      Edit = No: GOSUB Identifier: SaveFix$ = LastFix$: IF Max = 1 THEN GOTO Enroute
      IF Id < 3 THEN GOSUB NavFix: LastFix$ = FormatId(Id$) ELSE LastFix$ = Text$: GOSUB GetIntersection
      IF Max > 1 THEN Flight$ = Flight$ + Fix$: OKtoEnd = Yes ELSE LastFix$ = SaveFix$
      GOTO Enroute
   END IF

RouteMenu:
   Printer = No: Heading$ = Departure$ + " to " + Destination$: ScreenHeading
   LOCATE 9, 35: COLOR 0, C(1): PRINT " 1 "; : COLOR CI(1), 0: PRINT "  VIEW"
   LOCATE 11, 35: COLOR 0, C(3): PRINT " 2 "; : COLOR CI(3), 0: PRINT "  PRINT"
   LOCATE 13, 35: COLOR 0, C(4): PRINT " 3 "; : COLOR CI(4), 0: PRINT "  REVERSE"
   LOCATE 15, 35: COLOR 0, C(2): PRINT " 4 "; : COLOR CI(2), 0: PRINT "  SAVE"
   LOCATE 19, 35: COLOR 0, C(0): PRINT " 0 "; : COLOR CI(0), 0: PRINT "  MENU"
   DO
      Menu 4: Hue = 1: ON K GOTO RouteDisplay, PrintSetup, Reverse, FlightSave: GOTO MainMenu

Reverse:
      SWAP Departure$, Destination$: Heading$ = Departure$ + " to " + Destination$: Direction = -Direction
      COLOR 15: Wipe 2, 1: LOCATE 2, INT((82 - LEN(Heading$)) / 2): PRINT Heading$
   LOOP

PrintSetup:
   Printer = Yes: Hue = 3: ScreenHeading: LOCATE 13, 22: PRINT "Prepare Printer {Top-of-form & On-line}"
   Escape "printing": Continue 15: IF K$ = Esc$ THEN GOTO RouteMenu

RouteDisplay:
   CALL NavFile: ScreenHeading: DistWarnFlag = Yes: IF Printer THEN Escape "printing" ELSE Escape "listing"
   LOCATE 14, 35: PRINT "* "; : COLOR CI(Hue) + 16: PRINT "STANDBY"; : COLOR CI(Hue): PRINT " *"
   Pointer = 1: Legs = 0: TotalDistance& = 0: FlightPointer$ = "": GOSUB Pointers: DistFlag = Yes
   DO
      FrLat# = ToLat#: FrLng# = ToLng#: FrVar# = ToVar#: GOSUB Pointers: IF Id = 5 THEN EXIT DO
      GOSUB DistanceAndCourse: TotalDistance& = TotalDistance& + D# * RS#
   LOOP: DistFlag = No: IF INKEY$ = Esc$ THEN GOTO QuitDisplay
   PrMsg$ = STR$(TotalDistance&) + " NM": Distance$ = "Total Distance:" + PrMsg$
   IF Printer THEN
      ON ERROR GOTO RoutePrintError: LPRINT PrInit$; SPACE$(CINT((80 - LEN(Heading$)) / 2)); Heading$
      LPRINT SPACE$(CINT((79 - LEN(PrMsg$)) / 2)); PrMsg$
      LPRINT SPC(30); STRING$(20, DoubleLine$): LPRINT : ON ERROR GOTO 0
   END IF: Wipe 14, 35: Wipe 4, 1: LOCATE 4, CINT((81 - LEN(Distance$)) / 2): PRINT Distance$
   Row = 6: Pointer = 1: IF Direction < 0 THEN Pointer = ASC(RIGHT$(FlightPointer$, 1))
   GOSUB FlightLegData: FrLat# = ToLat#: FrLng# = ToLng#: FrVar# = ToVar#
   IF MID$(Flight$, Pointer, 1) = "4" THEN GOSUB TerminalWayPointData
   LegIdFlag = Yes: GOSUB LegDisplay
   FirstLeg = 2 * -(Direction > 0) + (Legs - 1) * -(Direction < 0)
   LastLeg = Legs * -(Direction > 0) + -(Direction < 0): WPs = 0
   FOR Leg = FirstLeg TO LastLeg STEP Direction
      WaypointFlag = No: Pointer = ASC(MID$(FlightPointer$, Leg, 1))
      IF MID$(Flight$, Pointer, 1) = "2" THEN 'Fix is a Waypoint
         WPs = WPs + 1
      ELSE
         GOSUB FlightLegData: GOSUB DistanceAndCourse: TC# = A#
         LegDistance = D# * RS#: TotalDistance& = TotalDistance& - LegDistance: WaypointLegDist = 0
         WHILE WPs 'Calculate the waypoints
            Pointer = ASC(MID$(FlightPointer$, Leg - WPs * Direction, 1)): WPDFlag = Yes
            WPs = WPs - 1: Id = 2: GOSUB Coordinates: GOSUB DistanceAndCourse
            B# = A# - TC#: IF ABS(B#) > PI# THEN B# = B# - D360# * SGN(B#)
            IF ABS(B#) <= D90# THEN
               WaypointsDist = ASin(SIN(D#) * SIN(D90# - ABS(B#))) * RS#
               IF WaypointsDist < LegDistance AND WaypointLegDist < WaypointsDist THEN
                  Distance$ = STR$(WaypointsDist - WaypointLegDist): WaypointLegDist = WaypointsDist
                  RemDist$ = STR$(TotalDistance& + LegDistance - WaypointLegDist): GOSUB FormatMC
                  LegIdFlag = No: GOSUB LegDisplay: R# = A# + 2 * D90# + SGN(B#) * (D90# - ABS(B#)) + ToVar#
                  IF R# < 0 THEN R# = R# + D360# ELSE IF R# > D360# THEN R# = R# - D360#
                  Radial = R# * 10 * R2D#: D# = SIN(D#) * SIN(ABS(B#))
                  Distance& = ASin(D#) * 10 * RS#: ToId$ = Identifier$: ToFreqPr$ = FreqPr$: FrVar# = ToVar#
                  WaypointFlag = Yes: GOSUB FormatRadial: LegIdFlag = Yes: Id = 2: Facility: GOSUB LegDisplay
               END IF
            END IF: WPDFlag = No
         WEND
         IF WaypointFlag THEN Pointer = ASC(MID$(FlightPointer$, Leg, 1)): GOSUB FlightLegData
         GOSUB FormatMC: Distance$ = STR$(LegDistance - WaypointLegDist): RemDist$ = STR$(TotalDistance&)
```

```
         LegIdFlag = No: GOSUB LegDisplay: FrLat# = ToLat#: FrLng# = ToLng#: FrVar# = ToVar#
         IF MID$(Flight$, Pointer, 1) = "4" THEN GOSUB TerminalWayPointData
         LegIdFlag = Yes: GOSUB LegDisplay
      END IF
   NEXT Leg: Wipe 25, 1
   IF NOT Printer THEN Continue 23: GOTO QuitDisplay
   COLOR C(Hue): LOCATE 23, 26: PRINT "Number of line feeds (3-9): "; : COLOR C(Hue) + 16: PRINT CHR$(22)
   LOCATE 25, 33: COLOR 0, C(Hue): PRINT " RETURN "; : COLOR C(Hue), 0: PRINT " to page";
   DO
      ON ERROR GOTO RoutePrintError: K$ = INPUT$(1)
      IF K$ = Rtn$ OR K$ = Esc$ THEN LPRINT CHR$(12); : GOTO QuitDisplay
   LOOP WHILE K$ < "3" OR K$ > "9": K = VAL(K$) - 3
   LPRINT : LPRINT SPC(20); STRING$(40, SingleLine$): LPRINT : FOR X = 1 TO K: LPRINT : NEXT
QuitDisplay:
   CLOSE : ON ERROR GOTO 0: GOTO RouteMenu

FormatMC:
   MC# = TC# + FrVar#: GOSUB CkMC: FrMC$ = "MC: " + A$: ToMC$ = ""
   IF FrVar# <> ToVar# THEN MC# = TC# + ToVar#: GOSUB CkMC: ToMC$ = "/" + A$
   RETURN
CkMC:
   IF MC# < 0 THEN MC# = MC# + D360# ELSE IF MC# > D360# THEN MC# = MC# - D360#
   A$ = RIGHT$(STR$(CINT(MC# * R2D# + 1000)), 3): RETURN

FormatRadial:
   A$ = STR$(Radial): GOSUB CkRadial: Radial$ = A$: A$ = STR$(Distance&): GOSUB CkRadial: DME$ = A$: RETURN
CkRadial:
   A$ = MID$(A$, 2, LEN(A$) - 2) + "." + RIGHT$(A$, 1): RETURN

Pointers:
   Id = VAL(MID$(Flight$, Pointer, 1)): IF Id = 5 THEN RETURN
   CoordinateFlag = VAL(MID$(Flight$, Pointer + 5, 1))
   IF Id = 4 THEN Pointer = Increment(Pointer, CoordinateFlag): GOTO Pointers
   Legs = Legs + 1: FlightPointer$ = FlightPointer$ + CHR$(Pointer)
   IF Id = 3 THEN Pointer = Pointer + 5: GOTO Intersection
   IF Id = 2 THEN Pointer = Increment(Pointer, CoordinateFlag): GOTO Pointers ELSE GOTO Coordinates

DistanceAndCourse: 'Calculate Distance Angle & Course
   D# = SIN(FrLat#) * SIN(ToLat#) + COS(FrLat#) * COS(ToLat#) * COS(FrLng# - ToLng#): D# = ACos(D#)
   IF DistFlag THEN
      IF D# > .3 AND DistWarnFlag THEN
         DistWarnFlag = No: LOCATE 4, 28: COLOR CI(0): PRINT "Distance Warning Message!"
         LOCATE 8, 11: PRINT "As the course continually changes along a Great Circle route"
         LOCATE 10, 22: PRINT "long legs may produce heading errors!": COLOR CI(Hue)
         Continue 19: Wipe 8, 1: Wipe 10, 22: IF K$ = Esc$ THEN GOTO QuitDisplay
      END IF: RETURN
   END IF: A# = ACos(SIN(FrLat#) * SIN(ToLat#) + COS(FrLat#) * COS(ToLat#))
   IF D# THEN A# = ASin(SIN(A#) / SIN(D#)) ELSE A# = D360#: RETURN
   A# = A# * SGN(ToLat# - FrLat#): IF FrLng# > ToLng# THEN A# = D90# - A# ELSE A# = A# + 3 * D90#
   IF ABS(FrLng# - ToLng#) > PI# THEN A# = A# - PI#
   RETURN

Intersection:
   GOSUB Coordinates: Text$ = MID$(Flight$, Pointer, 5): Strip: Radial$ = Text$
   Text$ = MID$(Flight$, Pointer + 5, 5): Strip: DME$ = Text$
   R# = VAL(Radial$) * D2R# - ToVar#: D# = VAL(DME$) / RS#: L# = ToLat#
   A# = SIN(D#) * COS(R#): ToLat# = ToLat# + ASin(A#)
   A# = SIN(D#) * SIN(R#) / COS((L# + ToLat#) / 2): ToLng# = ToLng# - ASin(A#)
   Pointer = Pointer + 10: ToFreqPr$ = "": RETURN

FlightLegData:
   Id = VAL(MID$(Flight$, Pointer, 1)): Facility
   IF Id < 2 THEN GOSUB Coordinates: ToId$ = Identifier$: ToFreqPr$ = FreqPr$: RETURN
   ToId$ = MID$(Flight$, Pointer + 1, 5): Pointer = Pointer + 5: GOTO Intersection

TerminalWayPointData:
   Id = 4: GOSUB Coordinates: GOSUB DistanceAndCourse: FrVar# = ToVar#: Distance& = D# * RS# * 10
   R# = A# + ToVar# + PI#: IF R# > D360# THEN R# = R# - D360#
   Radial = R# * R2D# * 10: Id = 4: GOTO FormatRadial

Coordinates:
   CoordinateFlag = VAL(MID$(Flight$, Pointer + 5, 1)): Identifier$ = MID$(Flight$, Pointer, 5)
   IF Id > 1 THEN MID$(Identifier$, 1, 1) = "0"
   IF CoordinateFlag THEN
      LSET Lat$ = MID$(Flight$, Pointer + 6, 8): LSET Lng$ = MID$(Flight$, Pointer + 14, 9)
      LSET Var$ = MID$(Flight$, Pointer + 23, 3): LSET Freq$ = MID$(Flight$, Pointer + 26, 7)
```

```
      Identifier$ = RIGHT$(Identifier$, 4)
   ELSE
      Ndx = INSTR(Index$, Identifier$)
      IF Ndx = 0 THEN
         Heading$ = "Nav File Error": Hue = 0: ScreenHeading: Facility: CLOSE
         LOCATE 11, 30: PRINT "Can't find the "; UCASE$(Fac$): LOCATE 13, 39: PRINT RIGHT$(Identifier$, 4)
         LOCATE 15, 33: PRINT "in the Nav File!": Continue 23: GOTO RouteMenu
      END IF: Rec = VAL(MID$(Index$, Ndx + 5, 3)): GET 1, Rec + 2: Identifier$ = Id$
   END IF 'Convert Degrees to Radians
   ToLat# = VAL(LEFT$(Lat$, 3)) + VAL(MID$(Lat$, 5, 2)) / 60
   ToLat# = (ToLat# + VAL(MID$(Lat$, 7, 2 - Format)) / (3600 / (1 + 5 * Format))) * D2R#
   ToLng# = VAL(LEFT$(Lng$, 4)) + VAL(MID$(Lng$, 6, 2)) / 60
   ToLng# = (ToLng# + VAL(MID$(Lng$, 8, 2 - Format)) / (3600 / (1 + 5 * Format))) * D2R#
   ToVar# = VAL(Var$) * D2R#: Pointer = Increment(Pointer, CoordinateFlag)
   Text$ = Freq$: Strip: FreqPr$ = Text$: IF LEN(Text$) = 0 THEN RETURN
   IF INSTR(FreqPr$, ".") THEN FreqPr$ = " " + FreqPr$ + " MHz" ELSE FreqPr$ = " (" + FreqPr$ + ")"
   RETURN

LegDisplay: 'Flight Leg Display Routine
   Row = Row + 1: IF INKEY$ = Esc$ THEN GOTO QuitDisplay
   IF Row < 22 THEN
      IF LegIdFlag THEN
         Msg$ = SPACE$(6) + FormatId(ToId$) + " " + Fac$ + ToFreqPr$: PrMsg$ = Msg$
         IF Id > 1 THEN
            IF Id > 2 THEN Msg$ = Msg$ + " * " + FormatId(Identifier$) + FreqPr$
            PrMsg$ = Msg$ + " " + Radial$ + PrDeg$ + " " + DME$ + " NM"
            Msg$ = Msg$ + " " + Radial$ + Degree$ + " " + DME$ + " NM"
         END IF: PrMsg$ = PrMsg$ + RemDist$: Msg$ = Msg$ + RemDist$
      ELSE
         Msg$ = SPACE$(8) + FrMC$ + Degree$: PrMsg$ = SPACE$(8) + FrMC$ + PrDeg$
         IF LEN(ToMC$) THEN Msg$ = Msg$ + ToMC$ + Degree$: PrMsg$ = PrMsg$ + ToMC$ + PrDeg$
         Text$ = "  Distance:" + Distance$ + " NM": Msg$ = Msg$ + Text$: PrMsg$ = PrMsg$ + Text$
         IF Leg < LastLeg - 1 OR WPDFlag THEN RemDist$ = "  RemDist:" + RemDist$ + " NM" ELSE RemDist$ = ""
      END IF: IF Printer THEN ON ERROR GOTO RoutePrintError:  LPRINT PrMsg$: ON ERROR GOTO 0
      LOCATE Row, 1: PRINT Msg$: RETURN
   ELSE
      IF NOT Printer THEN Continue 23: IF K$ = Esc$ THEN GOTO QuitDisplay
   END IF: FOR Row = 6 TO 21: Wipe Row, 1: NEXT: Wipe 23, 1
   LOCATE 7, 1: PRINT Msg$: Row = 7: GOTO LegDisplay

NavDataEntry:
   Heading$ = "NAV DATA MENU": ScreenHeading: Routing = No
   LOCATE 8, 34: COLOR 0, C(1): PRINT " 1 "; : COLOR CI(1), 0: PRINT "  ADD/EDIT"
   LOCATE 10, 34: COLOR 0, C(2): PRINT " 2 "; : COLOR CI(2), 0: PRINT "  DELETE"
   LOCATE 12, 34: COLOR 0, C(3): PRINT " 3 "; : COLOR CI(3), 0: PRINT "  LIST"
   LOCATE 14, 34: COLOR 0, C(4): PRINT " 4 "; : COLOR CI(4), 0: PRINT "  PRINT"
   LOCATE 17, 34: COLOR 0, C(5): PRINT " 5 "; : COLOR CI(5), 0: PRINT "  FORMAT"
   LOCATE 20, 34: COLOR 0, C(0): PRINT " 0 "; : COLOR CI(0), 0: PRINT "  MAIN MENU"
   Menu 5: IF K = 0 THEN GOTO MainMenu
   IF K = 5 THEN Text$ = "  RETURN TO NAV DATA MENU": LatLngFormat: GOTO NavDataEntry
   IF K = 2 THEN GOTO DeleteNavData ELSE IF K > 2 THEN Printer = K - 3: GOTO ListNavData

'Add/Edit the Nav File
   CALL NavFile: Hue = 1: Heading$ = "ADD/EDIT NAV DATA": ScreenHeading: FormatDisplay
NavAddEdit:
   COLOR CI(Hue): LOCATE 5, 21: PRINT NRec; "Entries    ": AiportNavaid
   DataAdded = No: IF K THEN GOSUB SortNavData: GOTO NavDataEntry
   Changed = No: Escape "entry": LOCATE 5, 22: PRINT Fac$; "       "
   Edit = No: Editing = No: GOSUB Identifier: IF Max = 1 THEN GOTO EndAddEdit
   IF Ndx THEN
      RecNdx = Ndx + 1: NavRec: Navaid
      IF AF$ = "X" THEN
         COLOR CI(0) + 16: LOCATE 5, 22: PRINT "DELETED": COLOR CI(Hue): Continue 21: GOTO EndAddEdit
      END IF
      Editing = Yes: PageFlag = Yes: Escape "change": GOTO EditFacData
   END IF
   Rec = NRec
   IF NRec > 999 THEN 'Full Nav Data File
      LOCATE 17, 30: COLOR C(0), 0: PRINT "NAV DATA FILE IS FULL!"
      PRINT CHR$(7): Time 2: GOSUB SortNavData: GOTO NavDataEntry
   END IF

AddFacData:
   LOCATE 5, 22: PRINT "ADD "; Fac$; " ": PageFlag = No
   FOR I = 1 TO 4
      ON I GOSUB Latitude, Longtitude, Variation, RadioFreq: IF Max = 1 THEN GOTO EndAddEdit
```

```
   NEXT: DataAdded = Yes

EditFacData:
   LOCATE 5, 22: PRINT "EDIT "; Fac$; " "
   DO
      FieldChoice 5, 21, PageFlag: IF K$ = Rtn$ THEN Max = 2: GOTO EndAddEdit
      IF K$ = Esc$ THEN DataAdded = No: Max = 1: GOTO EndAddEdit
      IF K$ = "-" OR K$ = "+" THEN
         GOSUB StoreNavdata
         DO
            Index = Index - ASC(K$) + 44
            Index = -Index * (Index > 0 AND Index < NRec) - 0 * (Index > NRec) - (NRec - 1) * (Index < 0)
            Rec = VAL(MID$(Index$, Index * 9 + 6, 3)): GET 1, Rec + 2: RecNdx = Index * 9 + 2
         LOOP WHILE AF$ = "X": Id = VAL(AF$): Facility: Navaid: GOTO EditFacData
      ELSE
         Wipe 21, 1: LOCATE 25, 8: PRINT SPC(18); : LOCATE 25, 57: PRINT SPC(14); : Edit = Yes
         ON VAL(K$) GOSUB Edit1, Edit2, Edit3, Edit4, Edit5
      END IF
   LOOP
Edit1: Text$ = FormatId(Id$): GOTO GetId
Edit2: Text$ = STR$(VAL(Lat$)): Strip: GOTO GetLat
Edit3: Text$ = STR$(VAL(Lng$)): Strip: GOTO GetLng
Edit4: Text$ = Var$: MID$(Text$, 1, 1) = CHR$(87 + 18 * (VAL(Text$) < 0)): GOTO GetVar
Edit5: Text$ = Freq$: GOTO GetRadioFreq

EndAddEdit:
   IF Routing THEN RETURN ELSE GOSUB StoreNavdata: GOSUB Clean: GOTO NavAddEdit

Clean:
   Wipe 9, 1: Wipe 11, 1: Wipe 13, 1: Wipe 15, 1: Wipe 17, 1: Wipe 21, 1: RETURN

Identifier:
   LOCATE 9, 26: COLOR 0, C(Hue): PRINT " 1 "; : COLOR CI(Hue), 0: PRINT "  Identifier:"
GetId:
   Wipe 9, 43: Query 9, 43, 1 - Edit * 10, 3 + Id + (Id = 3): Strip
   IF Max > 1 THEN
      IF LEN(Text$) = 0 THEN
         IF Edit THEN Text$ = "* INVALID *": GOTO IdError
         Max = 1: RETURN
      END IF: IF Id = 3 THEN RETURN
      IF Id = 0 OR LEN(Text$) < 4 THEN Text$ = " " + LEFT$(Text$ + "   ", 3)
      Ndx$ = Text$: Ndx = INSTR(Index$, AF$ + Ndx$)
      IF Ndx = 0 THEN
         IF Deleting THEN Text$ = "* NOT FOUND *": GOTO IdError
         LSET Id$ = Text$
         IF Editing THEN DataAdded = Yes: Changed = Yes: MID$(Index$, RecNdx, 4) = Ndx$
      ELSE
        IF NOT Edit THEN RETURN ELSE IF Ndx$ <> Id$ THEN Text$ = "* DUPLICATE *": GOTO IdError
      END IF
   ELSE
      IF NOT Edit THEN RETURN
   END IF: LOCATE 9, 43: PRINT Id$: RETURN
IdError:
   Wipe 9, 43: LOCATE 9, 43: COLOR 0, C(0): PRINT Text$: BEEP: Time 1: Wipe 9, 43: GOTO Edit1

Latitude:
   LOCATE 11, 26: COLOR 0, C(Hue): PRINT " 2 "; : COLOR CI(Hue), 0: PRINT "  Latitude:"
GetLat:
   Wipe 11, 41: Query 11, 41, 3 - Edit * 10, 8 - F
   IF Max > 1 THEN
      Sign = SGN(VAL(Text$)): Text$ = STR$(ABS(VAL(Text$)) * 10000 + 1000000!)
      Text$ = MID$(Text$, 3, 2) + "." + MID$(Text$, 5, 4 - F): Length = 1
      IF ASC(Text$) = 48 THEN MID$(Text$, 1, 1) = " ": Length = 2
      Text$ = " " + Text$: MID$(Text$, Length, 1) = CHR$(45 + (Sign > -1) * 13)
      IF ABS(VAL(Text$)) > 90 OR VAL(MID$(Text$, 5, 2)) > 59 OR (F = 0 AND VAL(MID$(Text$, 7, 2)) > 59) THEN
         Wipe 11, 41: LOCATE 11, 41: COLOR 0, C(0): PRINT "* INVALID *"
         BEEP: Time 1: Wipe 11, 41: COLOR CI(Hue): GOTO Edit2
      END IF: LSET Lat$ = Text$: LatFormat: IF Edit THEN DataAdded = Yes
   ELSE
      IF Edit THEN LatFormat ELSE RETURN
   END IF: LOCATE 11, 41: PRINT Deg$: RETURN

Longtitude:
   LOCATE 13, 26: COLOR 0, C(Hue): PRINT " 3 "; : COLOR CI(Hue), 0: PRINT "  Longitude:"
GetLng:
   Wipe 13, 42: Query 13, 42, 3 - Edit * 10, 9 - F
```

```
   IF Max > 1 THEN
      Sign = SGN(VAL(Text$)): Text$ = STR$(ABS(VAL(Text$)) * 10000 + 10000000#)
      Text$ = MID$(Text$, 3, 3) + "." + MID$(Text$, 6, 4 - F): Length = 1
      IF ASC(Text$) = 48 THEN
         MID$(Text$, 1, 1) = " ": Length = 2
         IF MID$(Text$, 2, 1) = "0" THEN MID$(Text$, 2, 1) = " ": Length = 3
      END IF
      Text$ = " " + Text$: MID$(Text$, Length, 1) = CHR$(45 + (Sign > -1) * 13)
      IF ABS(VAL(Text$)) > 180 OR VAL(MID$(Text$, 6, 2)) > 59 OR (F = 0 AND VAL(MID$(Text$, 8, 2)) > 59) THEN
         Wipe 13, 42: LOCATE 13, 42: COLOR 0, C(0): PRINT "* INVALID *"
         BEEP: Time 1: Wipe 13, 42: COLOR CI(Hue), 0: GOTO Edit3
      END IF: LSET Lng$ = Text$: LngFormat: IF Edit THEN DataAdded = Yes
   ELSE
      IF Edit THEN LngFormat ELSE RETURN
   END IF: LOCATE 13, 42: PRINT Deg$: RETURN

Variation:
   LOCATE 15, 26: COLOR 0, C(Hue): PRINT " 4 "; : COLOR CI(Hue), 0: PRINT "  Magnetic Variation:"
GetVar:
   Wipe 15, 51: Query 15, 51, 4 - Edit * 10, 3
   IF Max > 1 THEN
      IF ASC(Text$) > 47 AND ASC(Text$) < 59 THEN Var = VAL(Text$) ELSE Var = VAL(RIGHT$(Text$, 2))
      IF INSTR(Text$, "E") OR INSTR(Text$, "-") THEN Var = -Var
      LSET Var$ = STR$(Var): VarFormat: IF Edit THEN DataAdded = Yes
   ELSE
      IF Edit THEN VarFormat ELSE RETURN
   END IF: LOCATE 15, 51: PRINT Deg$: RETURN

RadioFreq:
   LOCATE 17, 26: COLOR 0, C(Hue): PRINT " 5 "; : COLOR CI(Hue), 0: PRINT "  Radio Frequency:"
GetRadioFreq:
   Query 17, 48, 0 - Edit * 10, 7: IF K$ = Esc$ AND NOT Edit THEN RETURN
   IF K$ <> Esc$ THEN LSET Freq$ = Text$: DataAdded = Yes
   Max = 2: LOCATE 17, 48: PRINT Freq$: RETURN

StoreNavdata:
   IF DataAdded THEN
      PUT 1, Rec + 2: IF Changed THEN Sort = Yes
      IF NOT Editing THEN
         Index$ = Index$ + AF$ + Ndx$ + RIGHT$(STR$(Rec + 1000), 3) + CHR$(254)
         GET 1, 1: NRec = Rec + 1: LSET Rec$ = MKI$(NRec): PUT 1, 1: GET 1, Rec + 2: Sort = Yes
      END IF
   END IF: RETURN

SortNavData:
   IF Sort THEN
      Heading$ = "SORTING THE NAV DATA": ScreenHeading: COLOR CI(4): LOCATE 12, 34: PRINT "Records sorted"
      COLOR C(Hue), 0: LOCATE 16, 2: PRINT "1 "; : COLOR 0, C(Hue): PRINT STRING$(76 - LEN(STR$(NRec)), 8);
      COLOR C(Hue), 0: PRINT STR$(NRec): Escape "sorting": Length = 76 - LEN(STR$(NRec)): COLOR CI(4), C(Hue)
      Begin = LEN(Index$) - 17: Limit = 1: Sorted = No: SortedPortion# = Length / (Begin + 9)
      DO
         Sorting = No: LOCATE 16, 4: PRINT STRING$(Limit * SortedPortion#, 8)
         FOR X = Begin TO Limit STEP -9: Y = X + 9
            IF ASC(MID$(Index$, X + 2, 1)) > 64 AND ASC(MID$(Index$, Y + 2, 1)) < 65 THEN GOTO SkipSort
            IF ASC(MID$(Index$, X + 2, 1)) < 65 AND ASC(MID$(Index$, Y + 2, 1)) > 64 THEN GOTO DoSort
            IF MID$(Index$, X + 2, 3) = MID$(Index$, Y + 2, 3) THEN
               IF MID$(Index$, X, 2) > MID$(Index$, Y, 2) THEN GOTO DoSort ELSE GOTO SkipSort
            END IF
            IF MID$(Index$, X + 2, 3) > MID$(Index$, Y + 2, 3) THEN
DoSort:
               Temp$ = MID$(Index$, X, 8): MID$(Index$, X, 8) = MID$(Index$, Y, 8)
               MID$(Index$, Y, 8) = Temp$: Sorting = Y: Sorted = Yes
            END IF
SkipSort:
            IF INKEY$ = Esc$ THEN Sorting = No: Sorted = No: EXIT FOR
         NEXT: Limit = Sorting
      LOOP WHILE Sorting: LOCATE 16, 4: PRINT STRING$(Length, 8)
      OPEN "NavFile.NDX" FOR OUTPUT AS 2: PRINT #2, Index$
   END IF: CLOSE : Sort = No: RETURN

DeleteNavData:
   CALL NavFile: Hue = 2: Heading$ = "DELETE NAV DATA": ScreenHeading
   Deletions = No: Edit = No: IF NRec = 0 THEN GOTO EmptyFile
   DO
      Wipe 5, 22: COLOR CI(0) + 16: LOCATE 5, 22: PRINT "DELETE": COLOR CI(Hue)
      LOCATE 5, 52 - LEN(STR$(NRec - Deleted)): PRINT NRec - Deleted; "Entries": AiportNavaid
```

```
      IF K THEN 'Pack & Reindex
         IF Deletions THEN
            Heading$ = "PACK NAV DATA": ScreenHeading: Escape "packing": Deletions = No
            LOCATE 14, 36: PRINT "Packing": Temp$ = "": NewRec = 0: TempNavFile
            FOR Rec = 0 TO NRec - 1
               GET 1, Rec + 2
               IF AF$ <> "X" THEN
                  LSET Dat$ = Old$: PUT 2, NewRec + 2: Temp$ = Temp$ + AF$ + Id$
                  Temp$ = Temp$ + RIGHT$("00" + RIGHT$(STR$(NewRec), LEN(STR$(NewRec)) - 1), 3) + CHR$(254)
                  NewRec = NewRec + 1: LOCATE 14, 43: PRINT NewRec
               END IF: IF INKEY$ = Esc$ THEN CLOSE : KILL "TempNav.dat": GOTO NavDataEntry
            NEXT: GET 2, 1: LSET Dat$ = MKI$(NewRec): PUT 2, 1: CLOSE
            KILL "NavFile.DAT": NAME "TempNav.DAT" AS "NavFile.DAT": Deleted = 0
            Index$ = Temp$: OPEN "NavFile.NDX" FOR OUTPUT AS 2: PRINT #2, Index$
         END IF: CLOSE : GOTO NavDataEntry
      END IF: Escape "": LOCATE 5, 29: PRINT Fac$: Deleting = Yes: GOSUB Identifier: Deleting = No
      IF Max > 1 THEN
         CALL NavRec: Navaid: Text$ = FormatId(Id$): Strip
         LOCATE 21, 18: PRINT "Are you sure you want to "; : COLOR CI(0): PRINT "delete ";
         COLOR CI(Hue): PRINT Text$; : PRINT " (Y/N)? "; : COLOR C(Hue): PRINT CHR$(222);
         COLOR 16, C(Hue): PRINT "N"; : COLOR C(Hue), 0: PRINT CHR$(221): COLOR CI(Hue)
         DO
            K$ = UCASE$(INPUT$(1)): IF K$ = Esc$ THEN EXIT DO
            IF K$ = "Y" THEN
               Deletions = Yes: IF AF$ <> "X" THEN Deleted = Deleted + 1: LSET AF$ = "X"
            ELSEIF K$ = Rtn$ OR K$ = "N" THEN
               K$ = "Y": IF AF$ = "X" THEN LSET AF$ = RIGHT$(STR$(Id), 1): Deleted = Deleted - 1
               IF Deleted < 0 THEN Deleted = 0
            END IF: IF K$ = "Y" THEN PUT 1, Rec + 2
         LOOP WHILE K$ <> "Y"
      END IF: GOSUB Clean
   LOOP

ListNavData:
   CALL NavFile: Hue = Printer + 3: Msg$ = MID$("LISTPRINT", 1 + 4 * Printer, 4 + Printer)
   Heading$ = Msg$ + " NAV DATA": ScreenHeading: IF NRec = 0 THEN GOTO EmptyFile
   Text$ = STR$(NRec) + " Entries ": Col = 1 + INT(80 - LEN(Text$)) / 2: LOCATE 6, Col: PRINT Text$
   IF Printer THEN LOCATE 9, 21: PRINT "Prepare Printer {top-of-form & on-line}"
   LOCATE 11 + Printer * 2, 30: COLOR 0, C(Hue): PRINT " RETURN "; : COLOR CI(Hue), 0: PRINT "  All Nav Data"
   LOCATE 13 + Printer * 2, 35: COLOR 0, C(Hue): PRINT " 1 "; : COLOR CI(Hue), 0: PRINT "  Navaids"
   LOCATE 15 + Printer * 2, 35: COLOR 0, C(Hue): PRINT " 2 "; : COLOR CI(Hue), 0: PRINT "  Airports": Escape ""
   LOCATE 20 + Printer, 30: PRINT Msg$; " which Nav Data? "; : COLOR CI(Hue) + 16: PRINT CHR$(22)
   DO
      K$ = INPUT$(1): IF K$ = Esc$ THEN CLOSE : GOTO NavDataEntry
   LOOP WHILE K$ <> Rtn$ AND K$ <> "1" AND K$ <> "2"
   Id = VAL(K$) - 1: AF = Id: IF Id < 0 THEN Fac$ = "NAV" ELSE Facility
   Top$ = Fac$ + " DATA": Heading$ = Msg$ + " " + Top$: Rec = 0: Lines = 15 + Printer * 35
   IF Printer THEN
      Pages = INT(NRec / 50 + .99): Degree$ = PrDeg$: Page = 1: ScreenHeading
      LOCATE 14, 37: COLOR CI(Hue) + 16: PRINT "PRINTING": Escape "printing"
   END IF: Hdg$ = "Facility         Latitude        Longitude       Variation    Frequency"
OutputHeading:
   IF Printer THEN
      ON ERROR GOTO NavDataPrintError: LPRINT PrInit$; SPACE$((80 - LEN(Top$)) / 2); Top$
      LPRINT TAB(4); "Printed: "; MID$("  JanFebMarAprMayJunJulAugSepOctNovDec", VAL(LEFT$(DATE$, 2)) * 3, 3);
      LPRINT " "; MID$(DATE$, 4, 2); ", "; RIGHT$(DATE$, 4); TAB(71 - LEN(STR$(Pages)) - LEN(STR$(Page)));
      LPRINT SPACE$((-3 - LEN(STR$(Pages))) * (AF > -1)); "Page"; Page; : IF AF < 0 THEN LPRINT "of"; Pages
      LPRINT TAB(4); STRING$(74, DoubleLine$): LPRINT TAB(7); Hdg$
      LPRINT TAB(4); STRING$(74, SingleLine$): LPRINT
   ELSE
      CALL ScreenHeading: LOCATE 4, 7: PRINT Hdg$
      LOCATE 5, 4: PRINT STRING$(74, 196): Escape "listing": LOCATE 7, 1
   END IF: LineNo = 0
LineOutput:
   IF INKEY$ = Esc$ OR Rec = NRec THEN GOTO QuitListing
   GET 1, VAL(MID$(Index$, Rec * 9 + 6, 3)) + 2: Rec = Rec + 1: Id = VAL(AF$) + (AF$ = "X")
   IF AF < 0 OR AF = Id THEN Facility ELSE GOTO LineOutput
   Text$ = SPACE$(-5 * (AF > -1) + (AF = 0)) + Id$: IF AF < 0 THEN Text$ = Text$ + " - " + Fac$
   Text$ = Text$ + SPACE$(4 - 5 * (AF > -1) + Format - (Id = 0)): LatFormat
   Text$ = Text$ + Deg$ + SPACE$(4 + 2 * Format): LngFormat: LineNo = LineNo + 1
   Text$ = Text$ + Deg$ + SPACE$(7 + Format): VarFormat: Text$ = Text$ + Deg$ + "        " + Freq$
   IF Printer THEN LPRINT TAB(4); Text$ ELSE PRINT TAB(4); Text$
   IF LineNo < Lines THEN GOTO LineOutput
   IF Rec < NRec THEN
      IF Printer THEN LPRINT CHR$(12);  ELSE Continue 23: IF K$ = Esc$ THEN CLOSE : GOTO NavDataEntry
      Page = Page + 1: GOTO OutputHeading
```

```
   END IF
QuitListing:
   IF Printer THEN LPRINT CHR$(12); : Degree$ = DegreeSymbol$ ELSE Continue 23
   CLOSE : ON ERROR GOTO 0: GOTO NavDataEntry

FlightLoad:
   Heading$ = "LOAD FLIGHT DATA": ScreenHeading: Escape "": FlightIndex
   IF K$ = Esc$ THEN GOTO MainMenu
LoadMenu:
   LOCATE 22, 31: COLOR 7, 0: PRINT "Load which flight? "; : COLOR 23: PRINT CHR$(22)
   IF K$ = "" OR K$ < "A" OR K$ > "Z" THEN
      DO
         K$ = UCASE$(INPUT$(1)): IF K$ = Esc$ OR K$ = Rtn$ THEN GOTO MainMenu
      LOOP WHILE K$ < "A" OR K$ > "Z"
   END IF: OPEN "FltData." + K$ FOR INPUT AS 2: INPUT #2, Id1$
   IF Id1$ = "EMPTY" THEN
      LOCATE 22, 50: COLOR 0, C(0): PRINT "* EMPTY *": Time 1
      CLOSE 2: K$ = "": COLOR 7, 0: Wipe 22, 49: GOTO LoadMenu
   END IF: ON ERROR GOTO 0
   INPUT #2, Id2$, Flight$: FlightDisplay: CLOSE
   IF K$ = Esc$ THEN GOTO MainMenu ELSE IF K$ = "N" THEN GOTO FlightLoad ELSE GOTO RouteMenu

FlightSave:
   Id1$ = Departure$: Id2$ = Destination$: IF Direction < 0 THEN SWAP Id1$, Id2$
   Heading$ = "SAVE FLIGHT " + Id1$ + " to " + Id2$: ScreenHeading
   Escape "": FlightIndex: ON ERROR GOTO 0: IF K$ = Esc$ THEN GOTO RouteMenu
   LOCATE 22, 28: COLOR 7, 0: PRINT "Store in which position? "; : COLOR 23: PRINT CHR$(22)
   IF K$ = "" OR K$ < "A" OR K$ > "Z" THEN
      DO
         K$ = UCASE$(INPUT$(1)): IF K$ = Esc$ OR K$ = Rtn$ THEN GOTO RouteMenu
      LOOP WHILE K$ < "A" OR K$ > "Z"
   END IF: COLOR 7, 0: LOCATE 22, 53: PRINT K$; : OPEN "FltData." + K$ FOR OUTPUT AS 2
   WRITE #2, Id1$, Id2$, Flight$: CLOSE : GOTO RouteMenu

PrinterCodes:
   Heading$ = "PRINTER CODES": ScreenHeading: LOCATE 6, 16: COLOR 0, 7
   PRINT " 1 "; : COLOR 7, 0: PRINT " Printer Initialization:": LOCATE 8, 19
   FOR I = 1 TO 24
      IF I = 13 THEN LOCATE 10, 19
      PRINT Zero(MID$(PrInit$, I, 1)); " ";
   NEXT: LOCATE 13, 16: COLOR 0, 7: PRINT " 2 "; : COLOR 7, 0: PRINT " Degree Symbol:": LOCATE 15, 19
   FOR I = 1 TO 24
      IF I = 13 THEN LOCATE 17, 19
      PRINT Zero(MID$(PrDeg$, I, 1)); " ";
   NEXT: LOCATE 20, 16: COLOR 0, 7: PRINT " 3 "; : COLOR 7, 0
   PRINT " Single Line ("; CHR$(196); "): "; : PRINT Zero(SingleLine$)
   LOCATE 20, 46: PRINT "Double Line ("; CHR$(205); "): "; : PRINT Zero(DoubleLine$): Escape ""

EnterPrCodes:
   Hue = 1: FieldChoice 3, 23, 0: Hue = 6: Wipe 23, 18: IF K = 0 THEN GOTO MainMenu
   LOCATE 23, 29: PRINT "ENTER DECIMAL ASCII CODES": COLOR 7, 0
   IF K = 3 THEN
      Text$ = "": K$ = STR$(ASC(SingleLine$)): IF VAL(K$) THEN Text$ = RIGHT$(K$, LEN(K$) - 1)
      Query 20, 37, 12, 3: IF K$ = Esc$ THEN GOTO PrinterCodes
      K$ = CHR$(VAL(Text$) MOD 256): LOCATE 20, 37: PRINT Zero(K$): SingleLine$ = K$
      Text$ = "": K$ = STR$(ASC(DoubleLine$)): IF VAL(K$) THEN Text$ = RIGHT$(K$, LEN(K$) - 1)
      Query 20, 63, 12, 3: IF K$ = Esc$ THEN GOTO PrinterCodes
      K$ = CHR$(VAL(Text$) MOD 256): LOCATE 20, 63: PRINT Zero(K$): DoubleLine$ = K$
   ELSE
      IF K = 1 THEN PR$ = PrInit$ ELSE PR$ = PrDeg$
      FOR I = 1 TO 24
         Row = 8 + (K - 1) * 7 - (I > 12) * 2: Col = 15 + I * 4 + (I > 12) * 48
         Text$ = "": K$ = STR$(ASC(MID$(PR$, I, 1)))
         IF VAL(K$) THEN Text$ = RIGHT$(K$, LEN(K$) - 1)
         Query Row, Col, 12, 3: IF K$ = Esc$ THEN GOTO PrinterCodes
         K$ = CHR$(VAL(Text$) MOD 256): LOCATE Row, Col
         PRINT Zero(K$): MID$(PR$, I, 1) = K$: IF Max = 1 THEN I = 24
      NEXT
   END IF: SysFile
   IF K = 1 THEN PrInit$ = PR$: LSET Pr1$ = PR$
   IF K = 2 THEN PrDeg$ = PR$: LSET Pr2$ = PR$
   IF K = 3 THEN LSET SL$ = SingleLine$: LSET DL$ = DoubleLine$
   PUT 3, 1: CLOSE 3: GOTO EnterPrCodes

ScreenInit:
   Heading$ = "SCREEN INITIALIZATION": ScreenHeading
```

```
      LOCATE 11, 30: COLOR 0, 7: PRINT " 1 "; : COLOR 7, 0: PRINT "  MONOCHROME"
   LOCATE 13, 30: COLOR 0, 7: PRINT " 2 "; : COLOR 7, 0: PRINT "  COLOR"
   LOCATE 21, 26: PRINT "Which monitor do you have? "; : COLOR 23: PRINT CHR$(22): COLOR 7, 0
   DO
      K$ = INPUT$(1): IF K$ = Rtn$ OR K$ = Esc$ THEN GOTO MainMenu
   LOOP WHILE K$ < "1" OR K$ > "2": LOCATE 21, 53: PRINT K$
   CALL SysFile: CFlg = VAL(K$) - 1: LSET CFlg$ = RIGHT$(STR$(CFlg), 1)
   PUT 3, 1: CLOSE : C$ = "777777": IF CFlg THEN C$ = "465312"
   FOR I = 0 TO 5: C(I) = VAL(MID$(C$, I + 1, 1)): IF CFlg THEN CI(I) = C(I) + 8 ELSE CI(I) = C(I)
   NEXT: GOTO MainMenu

EmptyFile:
   COLOR CI(Hue), 0: LOCATE 14, 29: PRINT "Nav Data File is empty!": CLOSE : Continue 24

FlightFileError:
   IF ERR = 53 THEN OPEN "FltData." + K$ FOR OUTPUT AS 2: WRITE #2, "EMPTY": CLOSE : RESUME
   ON ERROR GOTO 0

IndexFileError:
   IF ERR = 53 THEN OPEN "NavFile.NDX" FOR OUTPUT AS 2: PRINT #2, "": CLOSE : RESUME
   ON ERROR GOTO 0

NavDataPrintError:
   GOSUB PrinterError: RESUME NavDataEntry

RoutePrintError:
   GOSUB PrinterError: RESUME RouteMenu

PrinterError:
   IF ERR = 24 OR ERR = 25 THEN
      CALL ScreenHeading: COLOR C(0) + 15: LOCATE 14, 34: PRINT "PRINTER ERROR!"; CHR$(7)
      CLOSE : COLOR C(0): Continue 24: RETURN
   END IF: ON ERROR GOTO 0


'******* SUBROUTINES *******


FUNCTION ACos# (X#) STATIC
   IF X# = -1 THEN ACos# = PI#: EXIT FUNCTION
   ACos# = D90# - ASin(X#)
END FUNCTION

DEFINT A-Z
SUB AiportNavaid STATIC
   LOCATE 11, 35: COLOR 0, C(Hue): PRINT " 1 "; : COLOR CI(Hue), 0: PRINT "  Navaid"
   LOCATE 13, 35: COLOR 0, C(Hue): PRINT " 2 "; : COLOR CI(Hue), 0: PRINT "  Airport"
   LOCATE 17, 32: COLOR CI(Hue), 0: PRINT "Navaid or Airport? ";
   COLOR CI(Hue) + 16: PRINT CHR$(22): Escape ""
   DO
      K$ = INPUT$(1): : IF K$ = Esc$ OR K$ = Rtn$ THEN K = 1: EXIT SUB
   LOOP WHILE K$ <> "1" AND K$ <> "2": Id = VAL(K$) - 1: K = 0
   Wipe 11, 35: Wipe 13, 35: Wipe 17, 32: LSET AF$ = RIGHT$(STR$(Id), 1): Facility
END SUB

FUNCTION ASin# (X#) STATIC
   IF ABS(X#) = 1 THEN ASin# = PI# - SGN(X#) * D90#: EXIT FUNCTION
   ASin# = ATN(X# / SQR(ABS(1 - X# * X#)))
END FUNCTION

DEFINT A-Z
SUB Continue (Row) STATIC
   COLOR CI(Hue): LOCATE Row, 28: PRINT "Press any key to continue ";
   COLOR CI(Hue) + 16: PRINT CHR$(22); : COLOR CI(Hue): K$ = INPUT$(1): Wipe Row, 1: LOCATE 1, 1
END SUB

DEFINT A-Z
SUB Escape (Msg$) STATIC
   COLOR 0, 0: Wipe 25, 1: EscCol = 43 - CINT(LEN(Msg$) / 2): LOCATE 25, EscCol - 9
   COLOR 0, C(0): PRINT " ESC "; : COLOR CI(0), 0: PRINT " to abort " + Msg$; : COLOR CI(Hue), 0
END SUB

DEFINT A-Z
SUB Facility STATIC
   Fac$ = "Navaid": IF Id THEN Fac$ = "Airport": IF Id < 1 THEN Fac$ = "DELETED"
   IF Id > 1 THEN Fac$ = "Waypoint": IF Id = 3 THEN Fac$ = "Intersection"
```

```
'NOTES:
   '0 = NAVAID
   '1 = AIRPORT
   '2 = WAYPOINT
   '3 = INTERSECTION
   '4 = TERMINAL WAYPOINT
   '5 = END OF FLIGHT
END SUB

DEFINT A-Z
SUB FieldChoice (UpperLimit, Row, PageFlag) STATIC
   LOCATE Row, 18: COLOR CI(Hue), 0: PRINT "Enter Field to Change - or ";
   COLOR 0, C(Hue): PRINT " RETURN "; : COLOR CI(Hue), 0: PRINT " to Accept ";
   COLOR CI(Hue) + 16: PRINT CHR$(22); : K$ = ""
   IF PageFlag THEN
      LOCATE 25, 8: COLOR 0, C(Hue): PRINT " - "; : COLOR CI(Hue), 0: PRINT " Previous Entry";
      LOCATE 25, 57: COLOR 0, C(Hue): PRINT " + "; : COLOR CI(Hue), 0: PRINT " Next Entry";
   END IF: COLOR CI(Hue)
   WHILE K$ < "1" OR K$ > RIGHT$(STR$(UpperLimit), 1)
      K$ = INPUT$(1): K = VAL(K$): IF K$ = Esc$ OR K$ = Rtn$ THEN EXIT SUB
      IF PageFlag AND (K$ = "-" OR K$ = "+") THEN EXIT SUB
   WEND
END SUB

DEFINT A-Z
SUB FileCheck
   IF NRec = LEN(Index$) / 9 THEN EXIT SUB
   Heading$ = "FILE-INDEX ERROR": Hue = 0: ScreenHeading
   LOCATE 13, 26: PRINT "The NAV DATA file needs to be"
   LOCATE 15, 35: PRINT "Re-indexed!": Escape ""
   Continue 20: IF K$ = Esc$ THEN COLOR 7, 0: CLS : END ELSE IndexFile
END SUB

DEFINT A-Z
SUB FlightDisplay STATIC
   Heading$ = "FLIGHT ROUTE": Hue = 2: ScreenHeading: Escape "listing": Row = 7: Pointer = 1: Departure = Yes
ListNav:
   Id = VAL(MID$(Flight$, Pointer, 1)): Facility
   IF Id < 4 THEN
      K$ = INKEY$: IF K$ = Esc$ THEN EXIT SUB
      LOCATE Row, 26: Row = Row + 1
      IF Row = 23 THEN
         Msg$ = "Partial listing from " + Departure$: LOCATE 5, 28: PRINT Msg$
         Continue 23: IF K$ = Esc$ THEN EXIT SUB
         CALL ScreenHeading: LOCATE 5, 28: PRINT Msg$: Escape "listing"
         LOCATE 7, 26: PRINT LastFix$: Row = 9: LOCATE 8, 26
      END IF
   END IF
   ON Id GOTO Id1, Id1, Id3, Id4, Id5
Id1:
   Text$ = MID$(Flight$, Pointer + 1, 4): Strip: LastFix$ = " " + RIGHT$("    " + Text$, 4) + " " + Fac$
   PRINT LastFix$; : Pointer = Increment(Pointer, VAL(MID$(Flight$, Pointer + 5, 1))): GOTO DepartCk
Id3:
   Text$ = MID$(Flight$, Pointer + 1, 5): Strip: PRINT RIGHT$("     " + Text$, 5) + " " + Fac$
   Pointer = Increment(Pointer, VAL(MID$(Flight$, Pointer + 10, 1))) + 15
DepartCk:
   IF Departure THEN Departure$ = Text$: Departure = No
   GOTO ListNav
Id4:
   PRINT " * " + Fac$ + " " + MID$(Flight$, Pointer + 2, 3)
   Pointer = Increment(Pointer, VAL(MID$(Flight$, Pointer + 5, 1))): GOTO ListNav
Id5:
   Destination$ = Text$: Heading$ = Departure$ + " to " + Destination$
   Wipe 5, 28: LOCATE 5, 41 - LEN(Heading$) / 2: PRINT Heading$: Direction = 1
   LOCATE 23, 28: COLOR 7, 0: PRINT "Is this correct (Y/N)? "; CHR$(222);
   COLOR 16, 7: PRINT "Y"; : COLOR 7, 0: PRINT CHR$(221); : Escape "routing"
   DO
      K$ = UCASE$(INPUT$(1))
   LOOP WHILE K$ <> Esc$ AND K$ <> "N" AND K$ <> "Y" AND K$ <> Rtn$
END SUB

DEFINT A-Z
SUB FlightIndex STATIC
   ON ERROR GOTO FlightFileError
   FOR I = 0 TO 1: Col = I * 22 + 22: FOR Row = 7 TO 19: K$ = CHR$(13 * I + Row + 58)
      OPEN "FltData." + K$ FOR INPUT AS 2: INPUT #2, Text$
```

```
      IF Text$ <> "EMPTY" THEN INPUT #2, Temp$: Text$ = Text$ + " to " + Temp$
      CLOSE : LOCATE Row, Col: COLOR 0, C(2): PRINT CHR$(221) + K$ + CHR$(222);
      COLOR CI(2), 0: PRINT " "; Text$
      K$ = UCASE$(INKEY$): IF K$ = Esc$ THEN I = 1: Row = 19
      IF (K$ > "@" AND K$ < "[") THEN EXIT SUB
   NEXT: NEXT: ON ERROR GOTO 0
END SUB

DEFINT A-Z
SUB FormatDisplay STATIC
      A$ = "Format: DD" + DegreeSymbol$ + " MM"
      IF Format THEN A$ = A$ + ".d'" ELSE A$ = A$ + "' SS" + CHR$(34)
      LOCATE 5, (41 + Format * 2): PRINT A$
END SUB

FUNCTION FormatId$ (A$) STATIC
   Text$ = A$: Strip: FormatId$ = Text$
END FUNCTION

DEFINT A-Z
FUNCTION Increment (Pointer, CoordinateFlag) STATIC
   Increment = Pointer + 6 + CoordinateFlag * 27
'NOTES: Flight$ (Flight route data)
   'Navaid, Airport, or Waypoint -
      'Byte 1:                  Fix identifier (see Sub Facility)
      'Bytes 2 to 5:            Four letter identifier
      'Byte 6:                  Coordinate Flag (0 = in NavFile, 1 = in Flight$)
      'Bytes 7 to 33:           Coordinate data if stored in the Flight$
      '  Bytes 7 to 14:            Latitude (8 bytes - bytes 1 to 8 relative)
      '  Bytes 15 to 23:           Longtitude (9 bytes - bytes 9 to 17 relative)
      '  Bytes 24 to 26:           Variation (3 bytes - bytes 18 to 20 relative)
      '  Bytes 27 to 33:           Frequence (7 bytes - bytes 21 to 27 relative)
   'Intersection -
      'Byte 1:                  "3" for intersection
      'Bytes 2 to 6:            Five letter identifier
      'Bytes 7 to 11(38):       Reference Navaid Data
      '  Bytes 7 to 10:            Four letter identifier
      '  Byte 11:                  Coordinate Flag
      '  Bytes (12) to (38):       Coordinate data if stored in the Flight$
      'Bytes 12(39) to 16(43):  Radial (5 bytes - 000.1 to 360.0)
      'Bytes 17(44) to 21(48):  DME Distance (5 bytes - 0.0 to 999.9)
END FUNCTION

DEFINT A-Z
SUB IndexFile STATIC
   Hue = 5: ScreenHeading: LOCATE 14, 35: PRINT "Indexing": Rec = 0: Index$ = ""
   DO
      GET 1, Rec + 2
         IF ASC(Rec$) = 0 THEN
         EXIT DO
      ELSE
         Index$ = Index$ + AF$ + Id$ + RIGHT$("00" + RIGHT$(STR$(Rec), LEN(STR$(Rec)) - 1), 3) + CHR$(254)
         Rec = Rec + 1: LOCATE 14, 43: PRINT Rec
      END IF: GET 1, 1: LSET Rec$ = MKI$(Rec): PUT 1, 1: NRec = Rec
   LOOP: OPEN "NavFile.NDX" FOR OUTPUT AS 2: PRINT #2, Index$: CLOSE 2
END SUB

DEFINT A-Z
SUB IntroScreen STATIC
   COLOR C(4), C(4): CLS : COLOR 15: LOCATE 2, 40: PRINT CHR$(15)
   LOCATE 3, 37: COLOR CI(3): PRINT "IMKOKM;"
   LOCATE 4, 37: PRINT ": : : :"
   LOCATE 5, 18: COLOR 8: PRINT ".": LOCATE 5, 37: COLOR CI(3): PRINT "HKJMJK<"
   LOCATE 6, 16: COLOR 8: PRINT "DMmMD": LOCATE 6, 38: COLOR CI(3): PRINT ":   :"
   LOCATE 7, 38: PRINT ":   :"; : LOCATE 7, 57: COLOR CI(0): PRINT "^[[[": CHR$(16)
   LOCATE 8, 38: COLOR CI(3): PRINT ":   :              3"
   LOCATE 9, 38: PRINT ":   :               3"
   LOCATE 10, 18: PRINT "IMMMMMMMMMMMMMMMMMMMJMMMJMMMMMMMMMMMMMMMOMMMM;"
   LOCATE 11, 12: PRINT "IMMMMM<"; : COLOR CI(2): PRINT "           AIR NAVIGATION/*RNAV*           ";
      COLOR CI(3): PRINT "HMMMMMM;"
   LOCATE 12, 12: PRINT ":IMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM;  Jonesboro Eagles  :"
   LOCATE 13, 12: PRINT "::"; : COLOR CI(1): PRINT "        by Robert A. Tims         ";
      COLOR CI(3): PRINT ":  "; : COLOR CI(4): PRINT "Civil Air Patrol"; : COLOR CI(3): PRINT "  :"
   LOCATE 14, 12: PRINT "::"; SPC(34); ":    "; : COLOR CI(4)
      PRINT "Squadron  HQ"; : COLOR CI(3): PRINT "    :"
   LOCATE 15, 12: PRINT "::               "; : COLOR 15: PRINT "DDBDD"; SPC(14);
```

```
      COLOR CI(3): PRINT ": ZDBD? IMMMM; ZDBD? :"
   LOCATE 16, 12: PRINT "::                    "; : COLOR 15: PRINT "3"; SPC(16);
      COLOR CI(3): PRINT ": CDED4 :     : CDED4 :"
   LOCATE 17, 12: PRINT "::                      "; : COLOR 15: PRINT "U08"; SPC(15);
      COLOR CI(3): PRINT ": @DADY :     : @DADY :"
   LOCATE 18, 12: PRINT "::       "; : COLOR 15: PRINT "@DDDDDmDRDA"; : COLOR 0: PRINT "i";
      COLOR 15: PRINT "ADRDmDDDDDY     "; : COLOR CI(3): PRINT ":       :   x:       :"
   LOCATE 19, 12: PRINT "::                 "; : COLOR 8: PRINT "k     k"; SPC(13);
      COLOR CI(3): PRINT ":       :     :       :"
   LOCATE 20, 1: COLOR CI(5), C(5): PRINT "000000"; : COLOR 8, 7: PRINT STRING$(68, 178);
      COLOR CI(5), C(5): PRINT "000000"
   LOCATE 21, 1: PRINT "0000"; : COLOR 8, 7: PRINT STRING$(72, 178); : COLOR CI(5), C(5): PRINT "0000"
   LOCATE 22, 1: PRINT "00"; : COLOR 8, 7: PRINT STRING$(76, 178); : COLOR CI(5), C(5): PRINT "00"
   LOCATE 23, 1: COLOR CI(0), 0
      PRINT SPC(14); "Copyright 1989 ABS Logic, Inc.  All rights reserved."; SPC(14);
   LOCATE 24, 1: PRINT SPC(21); "1616 Alonzo Ave., Jonesboro, AR  72401"; SPC(21);
   LOCATE 25, 1: PRINT SPC(19); "Parts copyrighted 1985-1988 Microsoft Corp."; SPC(18); : Time 10
END SUB

DEFINT A-Z
SUB LatFormat STATIC
   L = INSTR(Lat$, "."): Deg$ = LEFT$(Lat$, L - 1) + Degree$ + " " + MID$(Lat$, L + 1, 2)
   IF Format THEN
      Deg$ = Deg$ + "." + MID$(Lat$, L + 3, 1) + "'"
   ELSE
      Deg$ = Deg$ + "' " + MID$(Lat$, L + 3, 2) + CHR$(34)
   END IF
END SUB

DEFINT A-Z
SUB LatLngFormat STATIC
Start:
   Hue = 5: Heading$ = "LAT/LNG FORMAT": ScreenHeading: COLOR CI(Hue)
   LOCATE 5, 21: PRINT NRec; "Entries": COLOR CI(Hue): FormatDisplay
   LOCATE 8, 25: COLOR 0, C(Hue): PRINT " 0 "; : COLOR CI(Hue), 0: PRINT Text$
   LOCATE 11, 25: COLOR 0, C(Hue): PRINT " 1 "; : COLOR CI(Hue), 0: PRINT "  DEGREES, MINUTES, & SECONDS"
   LOCATE 13, 30: PRINT "Format:  DD"; DegreeSymbol$; " MM' SS"; CHR$(34);
   LOCATE 14, 35: COLOR C(Hue): PRINT "Example:  90"; DegreeSymbol$; " 38' 47"; CHR$(34)
   LOCATE 15, 30: COLOR CI(Hue): PRINT "Entry form:  DD.MMDD";
   LOCATE 16, 35: COLOR C(Hue): PRINT "Example:  90.3847"
   LOCATE 19, 25: COLOR 0, C(Hue): PRINT " 2 "; : COLOR CI(Hue), 0: PRINT "  DEGREES, MINUTES, & TENTHS"
   LOCATE 21, 30: PRINT "Format:  DD"; DegreeSymbol$; " MM.d"
   LOCATE 22, 35: COLOR C(Hue): PRINT "Example:  90"; DegreeSymbol$; " 38.8'"
   LOCATE 23, 30: COLOR CI(Hue): PRINT "Entry form:  DD.MMd";
   LOCATE 24, 35: COLOR C(Hue): PRINT "Example:  90.388";
   DO
      K$ = INPUT$(1)
      IF K$ = Esc$ OR K$ = Rtn$ OR K$ = "0" THEN EXIT SUB
   LOOP WHILE K$ < "1" OR K$ > "2": K = VAL(K$) - 1
   Wipe 8, 1: FOR n = 19 TO 24: Wipe n - K * 8, 2: NEXT
   COLOR C(Hue), 0: Row = 19 - K * 4: LOCATE Row, 21
   PRINT "Your choice is format option #"; K$; " Y/N? ";
   PRINT CHR$(222); : COLOR 16, C(Hue): PRINT "Y"; : COLOR C(Hue), 0: PRINT CHR$(221)
   DO
      K$ = UCASE$(INPUT$(1))
      IF K$ = "N" OR K$ = Esc$ THEN Wipe Row, 21: GOTO Start
   LOOP WHILE K$ <> "Y" AND K$ <> Rtn$
   IF K <> Format THEN
      Save = Format: Format = K
      IF NRec > 0 THEN
         CALL NavFile: Heading$ = "REFORMATTING LAT/LNG": ScreenHeading
         LOCATE 5, 21: PRINT NRec; "Entries": FormatDisplay: Escape "": TempNavFile
         FIELD 2, 2 AS Trec$, 3 AS Dum$, 8 AS TLat$, 9 AS TLng$: LSET Trec$ = MKI$(NRec): PUT 2, 1
         FOR Rec = 1 TO NRec
            IF INKEY$ = Esc$ THEN CLOSE : KILL "TempNav.DAT": Format = Save: EXIT SUB
            LOCATE 14, 29: PRINT "Reformatting Record"; Rec: GET 1, Rec + 1: A$ = Lat$: B$ = Lng$
            A = VAL(MID$(A$, 7, 2)): B = VAL(MID$(B$, 8, 2))
            IF K THEN
               T = A / 6: MID$(A$, 7, 2) = RIGHT$(STR$(T), 1) + " "
               IF T = 10 THEN
                  A = VAL(MID$(A$, 5, 2)): T = A + 1
                  IF T = 60 THEN S$ = " 0" ELSE S$ = STR$(T)
                  MID$(A$, 5, 2) = RIGHT$("0" + RIGHT$(S$, LEN(S$) - 1), 2)
                  IF T = 60 THEN
                     A = ABS(VAL(A$)): Sign = SGN(VAL(A$)): A = A + 1: S$ = STR$(A)
                     MID$(A$, 1, 3) = RIGHT$(" " + CHR$(45 + (Sign > -1) * 13) + RIGHT$(S$, LEN(S$) - 1), 3)
```

```
                  END IF
               END IF
               T = B / 6: MID$(B$, 8, 2) = RIGHT$(STR$(T), 1) + " "
               IF T = 10 THEN
                  B = VAL(MID$(B$, 6, 2)): T = B + 1
                  IF T = 60 THEN S$ = " 0" ELSE S$ = STR$(T)
                  MID$(B$, 6, 2) = RIGHT$("0" + RIGHT$(S$, LEN(S$) - 1), 2)
                  IF T = 60 THEN
                     B = ABS(VAL(B$)): Sign = SGN(VAL(B$)): B = B + 1: S$ = STR$(B)
                     MID$(B$, 1, 4) = RIGHT$("  " + CHR$(45 + (Sign > -1) * 13) + RIGHT$(S$, LEN(S$) - 1), 4)
                  END IF
               END IF
            ELSE
               S$ = STR$(A * 6): MID$(A$, 7, 2) = RIGHT$("0" + RIGHT$(S$, LEN(S$) - 1), 2)
               S$ = STR$(B * 6): MID$(B$, 8, 2) = RIGHT$("0" + RIGHT$(S$, LEN(S$) - 1), 2)
            END IF: LSET Dat$ = Old$: LSET TLat$ = A$: LSET TLng$ = B$: PUT 2, Rec + 1
         NEXT: OPEN "NavFile.OLD" FOR RANDOM AS 3: CLOSE : KILL "NavFile.OLD"
         NAME "NavFile.DAT" AS "NavFile.OLD": NAME "TempNav.DAT" AS "NavFile.DAT"
      END IF: SysFile: LSET Format$ = RIGHT$(STR$(Format), 1): PUT 3, 1: CLOSE
   END IF
END SUB

DEFINT A-Z
SUB LngFormat STATIC
   L = INSTR(Lng$, "."): Deg$ = LEFT$(Lng$, L - 1) + Degree$ + " " + MID$(Lng$, L + 1, 2)
   IF Format THEN
      Deg$ = Deg$ + "." + MID$(Lng$, L + 3, 1) + "'"
   ELSE
      Deg$ = Deg$ + "' " + MID$(Lng$, L + 3, 2) + CHR$(34)
   END IF
END SUB

DEFINT A-Z
SUB Menu (UpperLimit) STATIC
   COLOR 7, 0: LOCATE 25, 32: PRINT "Which Function? "; CHR$(222);
   COLOR 16, 7: PRINT "0"; : COLOR 7, 0: PRINT CHR$(221); : K$ = ""
   WHILE K$ < "0" OR K$ > RIGHT$(STR$(UpperLimit), 1)
      K$ = INPUT$(1): IF K$ = Rtn$ THEN K$ = "0"
      IF K$ = Esc$ THEN
         CALL ScreenHeading: COLOR CI(0): K$ = "": LOCATE 15, 11
         PRINT "Quit the Air Navigation Program and return to DOS (Y/N)? ";
         COLOR C(0): PRINT CHR$(222); : COLOR 16, C(0): PRINT "N"; : COLOR C(0), 0: PRINT CHR$(221);
         WHILE K$ <> "N" AND K$ <> Rtn$ AND K$ <> Esc$
            K$ = INKEY$: IF K$ > "" THEN K$ = UCASE$(K$)
            IF K$ = "Y" THEN COLOR 7, 0: CLS : END
         WEND: K = 0: EXIT SUB
      END IF
   WEND: K = VAL(K$): LOCATE 25, 48: PRINT " "; K$; " ";
END SUB

DEFINT A-Z
SUB Navaid STATIC
   LOCATE 9, 43: COLOR CI(Hue), 0: PRINT Id$
   LOCATE 11, 26: LatFormat: COLOR 0, C(Hue)
   PRINT " 2 "; : COLOR CI(Hue), 0: PRINT "  Latitude: " + Deg$
   LOCATE 13, 26: LngFormat: COLOR 0, C(Hue)
   PRINT " 3 "; : COLOR CI(Hue), 0: PRINT "  Longitude: " + Deg$
   LOCATE 15, 26: VarFormat: COLOR 0, C(Hue)
   PRINT " 4 "; : COLOR CI(Hue), 0: PRINT "  Magnetic Variation: " + Deg$ + " "
   LOCATE 17, 26: COLOR 0, C(Hue)
   PRINT " 5 "; : COLOR CI(Hue), 0: PRINT "  Radio Frequency: " + Freq$
END SUB

DEFINT A-Z
SUB NavFile STATIC
   OPEN "NavFile.DAT" FOR RANDOM AS 1 LEN = 32
   FIELD 1, 1 AS AF$, 4 AS Id$, 8 AS Lat$, 9 AS Lng$, 3 AS Var$, 7 AS Freq$
   FIELD 1, 2 AS Rec$: GET 1, 1: NRec = CVI(Rec$): FileCheck
END SUB

DEFINT A-Z
SUB NavRec STATIC
   Rec = VAL(MID$(Index$, Ndx + 5, 3)): GET 1, Rec + 2: Index = INT(Ndx / 9)
END SUB

DEFINT A-Z
```

```
SUB Query (Row, Col, DataType, Length) STATIC
   Insert = 0
   IF INT(DataType / 10) THEN 'Check for Edit
      DataType = DataType - 10: LOCATE Row, Col
      IF DataType = 3 OR DataType = 5 THEN Flag = INSTR(Text$, "."): SignFlag = -(VAL(Text$) < 0)
      Text$ = Text$ + SPACE$(Length): Text$ = LEFT$(Text$, Length): Max = Length
      DO WHILE Max
         IF MID$(Text$, Max, 1) <> " " THEN EXIT DO ELSE Max = Max - 1
      LOOP
      IF Max THEN PRINT MID$(Text$, 1, Max);
      PRINT STRING$(Length - Max, "_"); : IF DataType = 4 THEN Flag = 1
      Max = Max + 1: Offset = 1: GOTO GetChar
   END IF
NewText:
   LOCATE Row, Col: Offset = 1: Flag = 0: Max = 1: SignFlag = 0
   PRINT STRING$(Length, 95); : Text$ = SPACE$(Length)
GetChar:
   IF Max = 1 THEN Msg$ = "abort" ELSE Msg$ = "erase"
   LOCATE 25, EscCol: COLOR CI(0), 0: PRINT Msg$; : COLOR CI(Hue), 0
   DO
      IF Length >= Offset AND (Flag = 0 OR DataType <> 3 OR Offset - Flag <> 5 - F) THEN
         S! = (TIMER - INT(TIMER)) * 2: S! = S! - INT(S!)
         IF S! >= .5 THEN LOCATE Row, Col + Offset - 1: PRINT CHR$(22 + 198 * Insert);
         IF S! < .5 THEN LOCATE Row, Col + Offset - 1: PRINT MID$(Text$, Offset, 1);
      END IF: K$ = INKEY$
   LOOP UNTIL K$ > CHR$(0)
   IF K$ = Rtn$ OR (K$ = Esc$ AND Max = 1) THEN 'Quit
      LOCATE 25, EscCol: COLOR CI(0), 0: PRINT "abort"; : COLOR CI(Hue), 0
      LOCATE Row, Col: PRINT Text$; : EXIT SUB
   END IF
   IF K$ = Esc$ THEN GOTO NewText
   IF K$ = CHR$(8) THEN 'Backspace
      IF Offset = 1 THEN GOTO GetChar ELSE Offset = Offset - 1: GOTO Delete
   END IF
   IF LEN(K$) = 2 THEN 'Check for extended key code
      Code = ASC(RIGHT$(K$, 1)) - 76: LOCATE Row, Col + Offset - 1: K$ = MID$(Text$, Offset, 1)
      IF Code = -5 THEN '[Home]
         IF Offset < Max THEN PRINT K$;  ELSE IF Offset < Length + 1 THEN PRINT "_";
         Offset = 1: GOTO GetChar
      END IF 'Then check for [End]
      IF Code = 3 AND Offset < Max THEN PRINT K$; : Offset = Max: GOTO GetChar
      IF Code = 6 AND DataType < 3 THEN Insert = Insert + (Insert = 1) - (Insert = 0): GOTO GetChar
      IF Code = 7 AND Offset < Max THEN
Delete:
         IF Offset = 1 THEN SignFlag = 0
         IF Flag = Offset THEN
            IF Flag <> Max - 1 THEN
               IF K$ = CHR$(8) THEN Offset = Offset + 1
               GOTO GetChar
            ELSE
               Flag = 0
            END IF
         END IF
         Temp$ = RIGHT$(Text$, Length - Offset)
         MID$(Text$, Offset, Length - Offset + 1) = Temp$ + " "
         LOCATE Row, Col + Offset - 1: Max = Max - 1
         IF Flag AND Offset < Flag THEN Flag = Flag - 1
         PRINT MID$(Temp$, 1, Max - Offset) + STRING$(Length + 1 - Max, "_"); : GOTO GetChar
      END IF 'Cursor [Left] or [Right]
      IF ABS(Code) <> 1 OR (Offset = Max AND Code = 1) THEN GOTO GetChar
      IF Offset < Max THEN PRINT K$;  ELSE IF Offset < Length + 1 THEN PRINT "_";
      IF (Offset > 1 OR Code = 1) AND (Offset < Length + 1 OR Code = -1) THEN Offset = Offset + Code
      GOTO GetChar
   END IF
   IF Offset > Length OR (K$ < " " OR K$ > CHR$(126)) OR K$ = CHR$(34) THEN GOTO GetChar
   ON DataType GOTO Caps, Number, LatLng, MagneticVariation, Decimal
KeepChar:
   LOCATE Row, Col + Offset - 1: PRINT K$;
   Temp$ = MID$(Text$, Offset, Length - Offset): MID$(Text$, Offset, 1) = K$
   IF Insert THEN
      Text$ = LEFT$(Text$, Offset) + Temp$: IF Max < Length + 1 THEN Max = Max + 1
      IF Max > 2 THEN LOCATE Row, Col: PRINT LEFT$(Text$, Max - 1)
   END IF
   Offset = Offset + 1: IF Offset - Max = 1 THEN Max = Offset
   GOTO GetChar
```

```
Caps: 'Data Type 1
   K$ = UCASE$(K$): GOTO KeepChar

Number: 'Data Type 2
   IF K$ < "0" OR K$ > "9" THEN GOTO GetChar ELSE GOTO KeepChar

LatLng: 'Data Type 3 - Insert disabled
   IF Offset = 1 THEN
      IF K$ = "-" OR K$ = "+" THEN SignFlag = 1: GOTO KeepChar
      IF SignFlag THEN GOTO GetChar
   END IF
   IF Flag = 0 AND K$ = "." AND Max - Offset < 6 - Format THEN Flag = Offset: GOTO KeepChar
   IF (Flag AND Offset - Flag > 4 - F) OR (Flag = 0 AND Offset > Length - 6 + Format + SignFlag) THEN
      GOTO GetChar
   END IF: IF K$ < "0" OR K$ > "9" THEN GOTO GetChar
   IF Offset = Flag THEN
      IF (Max = Offset + 1 AND Offset < Length - 5 + Format + SignFlag) THEN
         Flag = 0
      ELSE
         GOTO GetChar
      END IF
   END IF: GOTO KeepChar

MagneticVariation: 'Data Type 4 - Insert disabled & Middle-of-string backspace disabled before direction
   K$ = UCASE$(K$)
   IF (Flag = 0 OR Offset = Flag) THEN
      IF (K$ = "-" OR K$ = "+" OR K$ = "E" OR K$ = "W") THEN Flag = Offset: GOTO KeepChar
   END IF
   IF K$ > "/" AND K$ < ":" AND Flag <> Offset THEN
      IF (Flag = 1 OR Offset < Flag OR (Offset < 3 AND Flag = 0)) THEN GOTO KeepChar
   END IF: GOTO GetChar

Decimal: 'Data Type 5 - One Decimal place allowed - Insert disabled and backspace modified
   IF Flag = 0 AND K$ = "." AND Max - Offset < 3 THEN Flag = Offset: GOTO KeepChar
   IF (Flag AND Offset - Flag > 1) OR (Flag = 0 AND Offset > Length - 2) THEN GOTO GetChar
   IF K$ < "0" OR K$ > "9" THEN GOTO GetChar
   IF Offset = Flag THEN
      IF Offset < Length - 1 AND (Max = Offset + 1 OR Max < Length AND Offset < Length - 1) THEN
         Flag = 0
      ELSE
         GOTO GetChar
      END IF
   END IF: GOTO KeepChar
END SUB

DEFINT A-Z
SUB RouteScreen (WaypointFlag) STATIC
   LOCATE 9, 32: COLOR 0, C(Hue): PRINT " 1 "; : COLOR CI(Hue), 0: PRINT "  NAVAID"
   LOCATE 11, 32: COLOR 0, C(Hue): PRINT " 2 "; : COLOR CI(Hue), 0: PRINT "  AIRPORT"
   LOCATE 13, 32: COLOR 0, C(Hue): PRINT " 3 "; : COLOR CI(Hue), 0: PRINT "  INTERSECTION"
   IF WaypointFlag THEN
      LOCATE 15, 32: COLOR 0, C(Hue): PRINT " 4 "; : COLOR CI(Hue), 0: PRINT "  WAYPOINT"
      LOCATE 19, 36: PRINT "Next Fix? ";
   ELSE
      LOCATE 19, 32: PRINT "Departure Point? ";
   END IF: COLOR CI(Hue) + 16: PRINT CHR$(22): Escape "entry"
   DO
      K$ = INPUT$(1): IF K$ = Esc$ OR K$ = Rtn$ THEN EXIT SUB
   LOOP WHILE VAL(K$) < 1 OR VAL(K$) > 3 + WaypointFlag
   K = VAL(K$): Id = -(K - 1) * (K < 3) - 3 * (K = 3) - 2 * (K = 4)
END SUB

DEFINT A-Z
SUB ScreenHeading STATIC
      COLOR 0, 0, 0: CLS : LOCATE 1, 30: COLOR 15: PRINT "AIR NAVIGATION/*RNAV*"
      LOCATE 2, CINT((82 - LEN(Heading$)) / 2): PRINT Heading$
      LOCATE 3, 21: PRINT STRING$(40, 196): COLOR CI(Hue)
END SUB

DEFINT A-Z
SUB Strip STATIC
   Length = 0
   DO WHILE Length < LEN(Text$)
      IF MID$(Text$, Length + 1, 1) <> " " THEN Text$ = RIGHT$(Text$, LEN(Text$) - Length): EXIT DO
      Length = Length + 1
   LOOP
```

```
   DO WHILE LEN(Text$)
      IF MID$(Text$, LEN(Text$), 1) <> " " THEN EXIT DO
      Text$ = MID$(Text$, 1, LEN(Text$) - 1)
   LOOP
END SUB

SUB SysFile STATIC
   SHARED CFlg$, Format$, Pr1$, Pr2$, SL$, DL$
   OPEN "NAVSYS.DAT" FOR RANDOM AS 3
   FIELD 3, 1 AS CFlg$, 1 AS Format$, 24 AS Pr1$, 24 AS Pr2$, 1 AS SL$, 1 AS DL$
   IF LOF(3) = 0 THEN FIELD 3, 128 AS A$: LSET A$ = "00" + STRING$(48, 0) + "-=": PUT 3, 1
   GET 3, 1
END SUB

DEFINT A-Z
SUB TempNavFile
   OPEN "TempNav.DAT" FOR RANDOM AS 2: CLOSE 2: KILL "TempNav.DAT"
   OPEN "TempNav.DAT" FOR RANDOM AS 2 LEN = 32: FIELD 2, 32 AS Dat$: FIELD 1, 32 AS Old$
END SUB

SUB Time (Seconds) STATIC
   T! = TIMER: WHILE TIMER - (T! - T! * LEN(INKEY$)) < Seconds: WEND
END SUB

SUB VarFormat STATIC
   Deg$ = RIGHT$(STR$(100 + ABS(VAL(Var$))), 2) + Degree$ + " "
   IF ASC(Deg$) = 48 THEN MID$(Deg$, 1, 1) = " "
   IF VAL(Var$) < 0 THEN Deg$ = Deg$ + "E" ELSE Deg$ = Deg$ + "W"
END SUB

DEFINT A-Z
SUB Wipe (Row, Col) STATIC
   COLOR , 0: LOCATE Row, Col: PRINT SPC(80 - Col + (Row > 23));
END SUB

FUNCTION Zero$ (A$) STATIC
   Zero$ = RIGHT$("00" + RIGHT$(STR$(ASC(A$)), LEN(STR$(ASC(A$))) - 1), 3)
END FUNCTION
```

```
'AIR NAVIGATION/*RNAV* - QuickBasic Source Code - Copyright 1989 ROBERT A. TIMS & ABS Logic, Inc.
'1-27-89 Version Date - Program File Name: AIRNAV.BAS

'Subprogram and Function Declarations
   DECLARE FUNCTION ACos# (X#)
   DECLARE FUNCTION ASin# (X#)
   DECLARE FUNCTION FormatId$ (A$)
   DECLARE FUNCTION Increment% (A%, CoordinateFlag%)
   DECLARE FUNCTION Zero$ (A$)
   DECLARE SUB AiportNavaid ()
   DECLARE SUB Continue (Row%)
   DECLARE SUB Escape (Msg$)
   DECLARE SUB Facility ()
   DECLARE SUB FieldChoice (UpperLimit%, Row%, PageFlag%)
   DECLARE SUB FileCheck ()
   DECLARE SUB FlightDisplay ()
   DECLARE SUB FlightIndex ()
   DECLARE SUB FormatDisplay ()
   DECLARE SUB IndexFile ()
   DECLARE SUB IntroScreen ()
   DECLARE SUB LatFormat ()
   DECLARE SUB LatLngFormat ()
   DECLARE SUB LngFormat ()
   DECLARE SUB Menu (UpperLimit%)
   DECLARE SUB Navaid ()
   DECLARE SUB NavFile ()
   DECLARE SUB NavRec ()
   DECLARE SUB Query (Row%, Col%, DataType%, Length%)
   DECLARE SUB RouteScreen (WaypointFlag%)
   DECLARE SUB ScreenHeading ()
   DECLARE SUB Strip ()
   DECLARE SUB SysFile ()
   DECLARE SUB TempNavFile ()
   DECLARE SUB Time (Seconds!)
   DECLARE SUB VarFormat ()
   DECLARE SUB Wipe (Row%, Col%)

'Global and Constant Data
   DEFINT A-Z
   COMMON SHARED AF$, Cr$, Dat$, Deg$, Degree$, Departure$, Destination$, Esc$, Fac$, Flight$, Format$, Freq$
   COMMON SHARED Heading$, Id$, Id1$, Id2$, Index$, K$, Lat$, Lng$, Ndx$, Old$, Rec$, Rtn$, Text$, Var$
   COMMON SHARED Edit, EscCol, Format, Hue, Id, Index, K, Max, Ndx, NRec, Rec, Direction
   DIM SHARED C(6), CI(6): WIDTH "LPT1:", 255
   CONST PI# = 3.1415926535897793#
   CONST D90# = PI# / 2 'RADIANS IN A RIGHT ANGLE
   CONST D360# = 2 * PI# 'RADIANS FOR A CIRCLE
   CONST D2R# = PI# / 180 'CONSTANT TO CONVERT DEGREES TO RADIANS
   CONST R2D# = 180 / PI# 'CONSTANT TO CONVERT RADIANS TO DEGREES
   CONST RS# = R2D# * 60 'CONSTANT TO CONVERT EARTH DEGREES TO NM
   CONST No = 0, Yes = NOT No
   DegreeSymbol$ = CHR$(248): Degree$ = DegreeSymbol$: Esc$ = CHR$(27): Rtn$ = CHR$(13)

'System Parameters
   ON ERROR GOTO 0: SysFile
   CFlg = VAL(CFlg$): PrInit$ = Pr1$: PrDeg$ = Pr2$: Format = VAL(Format$)
   SingleLine$ = SL$: DoubleLine$ = DL$: CLOSE
   C$ = "777777": IF CFlg THEN C$ = "465312"
   FOR I = 0 TO 5: C(I) = VAL(MID$(C$, I + 1, 1)): CI(I) = C(I) + 8: NEXT
   C(6) = 7: CI(6) = 7: IF INKEY$ = "" THEN IntroScreen
   IF CFlg = 0 THEN FOR I = 0 TO 5: CI(I) = C(I): NEXT

'Index
   ON ERROR GOTO IndexFileError: OPEN "NavFile.NDX" FOR INPUT AS 2
   INPUT #2, Index$: CLOSE : ON ERROR GOTO 0
   CALL NavFile: IF LOF(1) = 0 THEN Text$ = "  GO TO MAIN MENU": LatLngFormat

MainMenu:
   CLOSE : Heading$ = "MAIN MENU": ScreenHeading
   LOCATE 8, 32: COLOR 0, C(1): PRINT " 1 "; : COLOR CI(1), 0: PRINT "  ROUTE ENTRY"
   LOCATE 10, 32: COLOR 0, C(2): PRINT " 2 "; : COLOR CI(2), 0: PRINT "  LOAD FLIGHT"
   LOCATE 12, 32: COLOR 0, C(3): PRINT " 3 "; : COLOR CI(3), 0: PRINT "  AIRNAV DATA"
   LOCATE 15, 32: COLOR 0, 7: PRINT " 4 "; : COLOR 7, 0: PRINT "  Printer Codes"
   LOCATE 17, 32: COLOR 0, 7: PRINT " 5 "; : COLOR 7, 0: PRINT "  Screen Setup"
   LOCATE 20, 32: COLOR 0, C(0): PRINT " 0 "; : COLOR CI(0), 0: PRINT "  QUIT"
   Menu 5: ON K GOTO RouteEntry, FlightLoad, NavDataEntry, PrinterCodes, ScreenInit
   IF K$ = "0" THEN COLOR 7, 0: CLS : END ELSE GOTO MainMenu
```

```
NavFix:
   IF Ndx THEN NavRec: Fix$ = Fix$ + Id$ + "0": RETURN
   Rec = NRec: Wipe 5, 1: FormatDisplay: DataAdded = No: GOSUB AddFacData: IF Max = 1 THEN RETURN
   IF Rec < 1000 THEN
      Wipe 21, 1: LOCATE 5, 22: COLOR CI(Hue): PRINT "FILE": LOCATE 21, 26
      PRINT "File with the Nav Data (Y/N)? "; : COLOR CI(Hue) + 16: PRINT CHR$(22): Escape "entry"
      DO
         K$ = UCASE$(INPUT$(1)): IF K$ = Esc$ THEN Max = 1: RETURN
      LOOP WHILE K$ <> "Y" AND K$ <> "N"
      IF K$ = "Y" THEN Fix$ = Fix$ + Id$ + "0": GOTO StoreNavdata
   END IF: Fix$ = Fix$ + Id$ + "1" + Lat$ + Lng$ + Var$ + Freq$: RETURN

GetIntersection:
   DO
      Temp$ = Fix$ + LEFT$(LastFix$ + "     ", 5): Msg$ = LastFix$ + " REFERENCE NAVAID"
         GOSUB DisplayFacility: Id = 0: Facility: LSET AF$ = "0"
      Edit = No: GOSUB Identifier: IF Max = 1 THEN RETURN
      Fix$ = "": GOSUB NavFix: IF K$ = Esc$ THEN RETURN
   LOOP WHILE Max = 1: Radial$ = "": Distance$ = ""
   Msg$ = LastFix$ + " INTERSECTION": GOSUB DisplayFacility: LOCATE 9, 30: PRINT "Reference Navaid:"; Id$
   FOR I = 1 TO 2
      ON I GOSUB GetRadial, GetDistance: IF Max = 1 THEN GOTO GetIntersection
   NEXT
   DO
      FieldChoice 2, 21, 0: IF K$ = Esc$ THEN Max = 1: RETURN
      IF K$ = Rtn$ THEN Fix$ = Temp$ + Fix$ + Radial$ + Distance$: RETURN
      Wipe 21, 1: Edit = Yes: ON VAL(K$) GOSUB EditRadial, EditDistance
   LOOP

GetRadial:
   LOCATE 14, 30: COLOR 0, C(Hue): PRINT " 1 "; : COLOR CI(Hue), 0: PRINT "  Radial:"
EditRadial:
   IF VAL(Radial$) = 0 THEN Text$ = "" ELSE Text$ = STR$(VAL(Radial$)): Strip
   Wipe 14, 43: Query 14, 43, 15, 5: IF Max = 1 THEN IF NOT Edit THEN RETURN ELSE Text$ = Radial$
   T! = VAL(Text$): T! = (T! * 10 MOD 3600) / 10: IF T! = 0 THEN T! = 360
   Text$ = STR$(T! + 1000): Text$ = RIGHT$(Text$, LEN(Text$) - 2): IF T! = INT(T!) THEN Text$ = Text$ + ".0"
   Radial$ = LEFT$(Text$ + "   ", 5): LOCATE 14, 43: PRINT Text$ + DegreeSymbol$: RETURN

GetDistance:
   LOCATE 16, 30: COLOR 0, C(Hue): PRINT " 2 "; : COLOR CI(Hue), 0: : PRINT "  Distance:"
EditDistance:
   IF VAL(Distance$) = 0 THEN Text$ = "" ELSE Text$ = STR$(VAL(Distance$)): Strip
   Wipe 16, 45: Query 16, 45, 15, 5: IF Max = 1 THEN IF NOT Edit THEN RETURN ELSE Text$ = Distance$
   T! = VAL(Text$): Text$ = STR$(T!): Text$ = RIGHT$(Text$, LEN(Text$) - 1)
   IF T! = INT(T!) THEN Text$ = Text$ + ".0"
   Distance$ = LEFT$(Text$ + "   ", 5): LOCATE 16, 45: PRINT Text$ + " NM": RETURN

TerminalWaypoint:
   IF Id = 1 THEN
      DO
         Msg$ = LastFix$ + " TERMINAL WAYPOINT": GOSUB DisplayFacility
         Id = 0: LSET AF$ = "0": Fix$ = "": Edit = No: GOSUB Identifier: IF Max = 1 THEN RETURN
         CALL Facility: GOSUB NavFix
      LOOP WHILE Max = 1: Flight$ = Flight$ + "4" + Fix$
   END IF: RETURN

DisplayFacility:
   CALL ScreenHeading: LOCATE 5, (80 - LEN(Msg$)) / 2: PRINT Msg$: Escape "entry": RETURN

RouteEntry:
   Editing = No: Routing = Yes: NavFile
Departure:
   Departure = Yes: OKtoEnd = No
   DO
      Hue = 1: Heading$ = "ROUTE PLANNING": ScreenHeading: RouteScreen 0
      IF K$ = Esc$ OR K$ = Rtn$ THEN GOTO MainMenu
      CALL Facility: Msg$ = "DEPARTURE " + UCASE$(Fac$): GOSUB DisplayFacility
      Fix$ = RIGHT$(STR$(Id), 1): LSET AF$ = Fix$: Edit = No: GOSUB Identifier
   LOOP WHILE Max = 1
   IF Id < 2 THEN GOSUB NavFix: LastFix$ = FormatId(Id$) ELSE LastFix$ = Text$: GOSUB GetIntersection
   IF Max = 1 THEN GOTO Departure ELSE Flight$ = Fix$: GOSUB TerminalWaypoint
Enroute:
   Departure = No: Msg$ = "LAST FIX WAS " + LastFix$: GOSUB DisplayFacility: RouteScreen 1
   IF K$ = Rtn$ AND OKtoEnd THEN
      GOSUB TerminalWaypoint: Flight$ = Flight$ + "5": GOSUB SortNavData: FlightDisplay
```

```
      IF K$ = Esc$ THEN GOTO MainMenu ELSE IF K$ = "N" THEN GOTO RouteEntry
   ELSE
      IF K$ = Rtn$ OR K$ = Esc$ THEN GOSUB SortNavData: GOTO MainMenu
      Fix$ = RIGHT$(STR$(Id), 1): Facility: Msg$ = "ENROUTE " + UCASE$(Fac$): GOSUB DisplayFacility
      LSET AF$ = Fix$: IF K = 4 THEN Id = 0: LSET AF$ = "0": Facility
      Edit = No: GOSUB Identifier: SaveFix$ = LastFix$: IF Max = 1 THEN GOTO Enroute
      IF Id < 3 THEN GOSUB NavFix: LastFix$ = FormatId(Id$) ELSE LastFix$ = Text$: GOSUB GetIntersection
      IF Max > 1 THEN Flight$ = Flight$ + Fix$: OKtoEnd = Yes ELSE LastFix$ = SaveFix$
      GOTO Enroute
   END IF

RouteMenu:
   Printer = No: Heading$ = Departure$ + " to " + Destination$: ScreenHeading
   LOCATE 9, 35: COLOR 0, C(1): PRINT " 1 "; : COLOR CI(1), 0: PRINT "  VIEW"
   LOCATE 11, 35: COLOR 0, C(3): PRINT " 2 "; : COLOR CI(3), 0: PRINT "  PRINT"
   LOCATE 13, 35: COLOR 0, C(4): PRINT " 3 "; : COLOR CI(4), 0: PRINT "  REVERSE"
   LOCATE 15, 35: COLOR 0, C(2): PRINT " 4 "; : COLOR CI(2), 0: PRINT "  SAVE"
   LOCATE 19, 35: COLOR 0, C(0): PRINT " 0 "; : COLOR CI(0), 0: PRINT "  MENU"
   DO
      Menu 4: Hue = 1: ON K GOTO RouteDisplay, PrintSetup, Reverse, FlightSave: GOTO MainMenu

Reverse:
      SWAP Departure$, Destination$: Heading$ = Departure$ + " to " + Destination$: Direction = -Direction
      COLOR 15: Wipe 2, 1: LOCATE 2, INT((82 - LEN(Heading$)) / 2): PRINT Heading$
   LOOP

PrintSetup:
   Printer = Yes: Hue = 3: ScreenHeading: LOCATE 13, 22: PRINT "Prepare Printer {Top-of-form & On-line}"
   Escape "printing": Continue 15: IF K$ = Esc$ THEN GOTO RouteMenu

RouteDisplay:
   CALL NavFile: ScreenHeading: DistWarnFlag = Yes: IF Printer THEN Escape "printing" ELSE Escape "listing"
   LOCATE 14, 35: PRINT "* "; : COLOR CI(Hue) + 16: PRINT "STANDBY"; : COLOR CI(Hue): PRINT " *"
   Pointer = 1: Legs = 0: TotalDistance& = 0: FlightPointer$ = "": GOSUB Pointers: DistFlag = Yes
   DO
      FrLat# = ToLat#: FrLng# = ToLng#: FrVar# = ToVar#: GOSUB Pointers: IF Id = 5 THEN EXIT DO
      GOSUB DistanceAndCourse: TotalDistance& = TotalDistance& + D# * RS#
   LOOP: DistFlag = No: IF INKEY$ = Esc$ THEN GOTO QuitDisplay
   PrMsg$ = STR$(TotalDistance&) + " NM": Distance$ = "Total Distance:" + PrMsg$
   IF Printer THEN
      ON ERROR GOTO RoutePrintError: LPRINT PrInit$; SPACE$(CINT((80 - LEN(Heading$)) / 2)); Heading$
      LPRINT SPACE$(CINT((79 - LEN(PrMsg$)) / 2)); PrMsg$
      LPRINT SPC(30); STRING$(20, DoubleLine$): LPRINT : ON ERROR GOTO 0
   END IF: Wipe 14, 35: Wipe 4, 1: LOCATE 4, CINT((81 - LEN(Distance$)) / 2): PRINT Distance$
   Row = 6: Pointer = 1: IF Direction < 0 THEN Pointer = ASC(RIGHT$(FlightPointer$, 1))
   GOSUB FlightLegData: FrLat# = ToLat#: FrLng# = ToLng#: FrVar# = ToVar#
   IF MID$(Flight$, Pointer, 1) = "4" THEN GOSUB TerminalWayPointData
   LegIdFlag = Yes: GOSUB LegDisplay
   FirstLeg = 2 * -(Direction > 0) + (Legs - 1) * -(Direction < 0)
   LastLeg = Legs * -(Direction > 0) + -(Direction < 0): WPs = 0
   FOR Leg = FirstLeg TO LastLeg STEP Direction
      WaypointFlag = No: Pointer = ASC(MID$(FlightPointer$, Leg, 1))
      IF MID$(Flight$, Pointer, 1) = "2" THEN 'Fix is a Waypoint
          WPs = WPs + 1
      ELSE
         GOSUB FlightLegData: GOSUB DistanceAndCourse: TC# = A#
         LegDistance = D# * RS#: TotalDistance& = TotalDistance& - LegDistance: WaypointLegDist = 0
         WHILE WPs 'Calculate the waypoints
            Pointer = ASC(MID$(FlightPointer$, Leg - WPs * Direction, 1)): WPDFlag = Yes
            WPs = WPs - 1: Id = 2: GOSUB Coordinates: GOSUB DistanceAndCourse
            B# = A# - TC#: IF ABS(B#) > PI# THEN B# = B# - D360# * SGN(B#)
            IF ABS(B#) <= D90# THEN
               WaypointsDist = ASin(SIN(D#) * SIN(D90# - ABS(B#))) * RS#
               IF WaypointsDist < LegDistance AND WaypointLegDist < WaypointsDist THEN
                  Distance$ = STR$(WaypointsDist - WaypointLegDist): WaypointLegDist = WaypointsDist
                  RemDist$ = STR$(TotalDistance& + LegDistance - WaypointLegDist): GOSUB FormatMC
                  LegIdFlag = No: GOSUB LegDisplay: R# = A# + 2 * D90# + SGN(B#) * (D90# - ABS(B#)) + ToVar#
                  IF R# < 0 THEN R# = R# + D360# ELSE IF R# > D360# THEN R# = R# - D360#
                  Radial = R# * 10 * R2D#: D# = SIN(D#) * SIN(ABS(B#))
                  Distance& = ASin(D#) * 10 * RS#: ToId$ = Identifier$: ToFreqPr$ = FreqPr$: FrVar# = ToVar#
                  WaypointFlag = Yes: GOSUB FormatRadial: LegIdFlag = Yes: Id = 2: Facility: GOSUB LegDisplay
               END IF
            END IF: WPDFlag = No
         WEND
         IF WaypointFlag THEN Pointer = ASC(MID$(FlightPointer$, Leg, 1)): GOSUB FlightLegData
         GOSUB FormatMC: Distance$ = STR$(LegDistance - WaypointLegDist): RemDist$ = STR$(TotalDistance&)
```

```
         LegIdFlag = No: GOSUB LegDisplay: FrLat# = ToLat#: FrLng# = ToLng#: FrVar# = ToVar#
         IF MID$(Flight$, Pointer, 1) = "4" THEN GOSUB TerminalWayPointData
         LegIdFlag = Yes: GOSUB LegDisplay
      END IF
   NEXT Leg: Wipe 25, 1
   IF NOT Printer THEN Continue 23: GOTO QuitDisplay
   COLOR C(Hue): LOCATE 23, 26: PRINT "Number of line feeds (3-9): "; : COLOR C(Hue) + 16: PRINT CHR$(22)
   LOCATE 25, 33: COLOR 0, C(Hue): PRINT " RETURN "; : COLOR C(Hue), 0: PRINT " to page";
   DO
      ON ERROR GOTO RoutePrintError: K$ = INPUT$(1)
      IF K$ = Rtn$ OR K$ = Esc$ THEN LPRINT CHR$(12); : GOTO QuitDisplay
   LOOP WHILE K$ < "3" OR K$ > "9": K = VAL(K$) - 3
   LPRINT : LPRINT SPC(20); STRING$(40, SingleLine$): LPRINT : FOR X = 1 TO K: LPRINT : NEXT
QuitDisplay:
   CLOSE : ON ERROR GOTO 0: GOTO RouteMenu

FormatMC:
   MC# = TC# + FrVar#: GOSUB CkMC: FrMC$ = "MC: " + A$: ToMC$ = ""
   IF FrVar# <> ToVar# THEN MC# = TC# + ToVar#: GOSUB CkMC: ToMC$ = "/" + A$
   RETURN
CkMC:
   IF MC# < 0 THEN MC# = MC# + D360# ELSE IF MC# > D360# THEN MC# = MC# - D360#
   A$ = RIGHT$(STR$(CINT(MC# * R2D# + 1000)), 3): RETURN

FormatRadial:
   A$ = STR$(Radial): GOSUB CkRadial: Radial$ = A$: A$ = STR$(Distance&): GOSUB CkRadial: DME$ = A$: RETURN
CkRadial:
   A$ = MID$(A$, 2, LEN(A$) - 2) + "." + RIGHT$(A$, 1): RETURN

Pointers:
   Id = VAL(MID$(Flight$, Pointer, 1)): IF Id = 5 THEN RETURN
   CoordinateFlag = VAL(MID$(Flight$, Pointer + 5, 1))
   IF Id = 4 THEN Pointer = Increment(Pointer, CoordinateFlag): GOTO Pointers
   Legs = Legs + 1: FlightPointer$ = FlightPointer$ + CHR$(Pointer)
   IF Id = 3 THEN Pointer = Pointer + 5: GOTO Intersection
   IF Id = 2 THEN Pointer = Increment(Pointer, CoordinateFlag): GOTO Pointers ELSE GOTO Coordinates

DistanceAndCourse: 'Calculate Distance Angle & Course
   D# = SIN(FrLat#) * SIN(ToLat#) + COS(FrLat#) * COS(ToLat#) * COS(FrLng# - ToLng#): D# = ACos(D#)
   IF DistFlag THEN
      IF D# > .3 AND DistWarnFlag THEN
         DistWarnFlag = No: LOCATE 4, 28: COLOR CI(0): PRINT "Distance Warning Message!"
         LOCATE 8, 11: PRINT "As the course continually changes along a Great Circle route"
         LOCATE 10, 22: PRINT "long legs may produce heading errors!": COLOR CI(Hue)
         Continue 19: Wipe 8, 1: Wipe 10, 22: IF K$ = Esc$ THEN GOTO QuitDisplay
      END IF: RETURN
   END IF: A# = ACos(SIN(FrLat#) * SIN(ToLat#) + COS(FrLat#) * COS(ToLat#))
   IF D# THEN A# = ASin(SIN(A#) / SIN(D#)) ELSE A# = D360#: RETURN
   A# = A# * SGN(ToLat# - FrLat#): IF FrLng# > ToLng# THEN A# = D90# - A# ELSE A# = A# + 3 * D90#
   IF ABS(FrLng# - ToLng#) > PI# THEN A# = A# - PI#
   RETURN

Intersection:
   GOSUB Coordinates: Text$ = MID$(Flight$, Pointer, 5): Strip: Radial$ = Text$
   Text$ = MID$(Flight$, Pointer + 5, 5): Strip: DME$ = Text$
   R# = VAL(Radial$) * D2R# - ToVar#: D# = VAL(DME$) / RS#: L# = ToLat#
   A# = SIN(D#) * COS(R#): ToLat# = ToLat# + ASin(A#)
   A# = SIN(D#) * SIN(R#) / COS((L# + ToLat#) / 2): ToLng# = ToLng# - ASin(A#)
   Pointer = Pointer + 10: ToFreqPr$ = "": RETURN

FlightLegData:
   Id = VAL(MID$(Flight$, Pointer, 1)): Facility
   IF Id < 2 THEN GOSUB Coordinates: ToId$ = Identifier$: ToFreqPr$ = FreqPr$: RETURN
   ToId$ = MID$(Flight$, Pointer + 1, 5): Pointer = Pointer + 5: GOTO Intersection

TerminalWayPointData:
   Id = 4: GOSUB Coordinates: GOSUB DistanceAndCourse: FrVar# = ToVar#: Distance& = D# * RS# * 10
   R# = A# + ToVar# + PI#: IF R# > D360# THEN R# = R# - D360#
   Radial = R# * R2D# * 10: Id = 4: GOTO FormatRadial

Coordinates:
   CoordinateFlag = VAL(MID$(Flight$, Pointer + 5, 1)): Identifier$ = MID$(Flight$, Pointer, 5)
   IF Id > 1 THEN MID$(Identifier$, 1, 1) = "0"
   IF CoordinateFlag THEN
      LSET Lat$ = MID$(Flight$, Pointer + 6, 8): LSET Lng$ = MID$(Flight$, Pointer + 14, 9)
      LSET Var$ = MID$(Flight$, Pointer + 23, 3): LSET Freq$ = MID$(Flight$, Pointer + 26, 7)
```

```
      Identifier$ = RIGHT$(Identifier$, 4)
   ELSE
      Ndx = INSTR(Index$, Identifier$)
      IF Ndx = 0 THEN
         Heading$ = "Nav File Error": Hue = 0: ScreenHeading: Facility: CLOSE
         LOCATE 11, 30: PRINT "Can't find the "; UCASE$(Fac$): LOCATE 13, 39: PRINT RIGHT$(Identifier$, 4)
         LOCATE 15, 33: PRINT "in the Nav File!": Continue 23: GOTO RouteMenu
      END IF: Rec = VAL(MID$(Index$, Ndx + 5, 3)): GET 1, Rec + 2: Identifier$ = Id$
   END IF 'Convert Degrees to Radians
   ToLat# = VAL(LEFT$(Lat$, 3)) + VAL(MID$(Lat$, 5, 2)) / 60
   ToLat# = (ToLat# + VAL(MID$(Lat$, 7, 2 - Format)) / (3600 / (1 + 5 * Format))) * D2R#
   ToLng# = VAL(LEFT$(Lng$, 4)) + VAL(MID$(Lng$, 6, 2)) / 60
   ToLng# = (ToLng# + VAL(MID$(Lng$, 8, 2 - Format)) / (3600 / (1 + 5 * Format))) * D2R#
   ToVar# = VAL(Var$) * D2R#: Pointer = Increment(Pointer, CoordinateFlag)
   Text$ = Freq$: Strip: FreqPr$ = Text$: IF LEN(Text$) = 0 THEN RETURN
   IF INSTR(FreqPr$, ".") THEN FreqPr$ = " " + FreqPr$ + " MHz" ELSE FreqPr$ = " (" + FreqPr$ + ")"
   RETURN

LegDisplay: 'Flight Leg Display Routine
   Row = Row + 1: IF INKEY$ = Esc$ THEN GOTO QuitDisplay
   IF Row < 22 THEN
      IF LegIdFlag THEN
         Msg$ = SPACE$(6) + FormatId(ToId$) + " " + Fac$ + ToFreqPr$: PrMsg$ = Msg$
         IF Id > 1 THEN
            IF Id > 2 THEN Msg$ = Msg$ + " * " + FormatId(Identifier$) + FreqPr$
            PrMsg$ = Msg$ + " " + Radial$ + PrDeg$ + " " + DME$ + " NM"
            Msg$ = Msg$ + " " + Radial$ + Degree$ + " " + DME$ + " NM"
         END IF: PrMsg$ = PrMsg$ + RemDist$: Msg$ = Msg$ + RemDist$
      ELSE
         Msg$ = SPACE$(8) + FrMC$ + Degree$: PrMsg$ = SPACE$(8) + FrMC$ + PrDeg$
         IF LEN(ToMC$) THEN Msg$ = Msg$ + ToMC$ + Degree$: PrMsg$ = PrMsg$ + ToMC$ + PrDeg$
         Text$ = "  Distance:" + Distance$ + " NM": Msg$ = Msg$ + Text$: PrMsg$ = PrMsg$ + Text$
         IF Leg < LastLeg - 1 OR WPDFlag THEN RemDist$ = "  RemDist:" + RemDist$ + " NM" ELSE RemDist$ = ""
      END IF: IF Printer THEN ON ERROR GOTO RoutePrintError:  LPRINT PrMsg$: ON ERROR GOTO 0
      LOCATE Row, 1: PRINT Msg$: RETURN
   ELSE
      IF NOT Printer THEN Continue 23: IF K$ = Esc$ THEN GOTO QuitDisplay
   END IF: FOR Row = 6 TO 21: Wipe Row, 1: NEXT: Wipe 23, 1
   LOCATE 7, 1: PRINT Msg$: Row = 7: GOTO LegDisplay

NavDataEntry:
   Heading$ = "NAV DATA MENU": ScreenHeading: Routing = No
   LOCATE 8, 34: COLOR 0, C(1): PRINT " 1 "; : COLOR CI(1), 0: PRINT "  ADD/EDIT"
   LOCATE 10, 34: COLOR 0, C(2): PRINT " 2 "; : COLOR CI(2), 0: PRINT "  DELETE"
   LOCATE 12, 34: COLOR 0, C(3): PRINT " 3 "; : COLOR CI(3), 0: PRINT "  LIST"
   LOCATE 14, 34: COLOR 0, C(4): PRINT " 4 "; : COLOR CI(4), 0: PRINT "  PRINT"
   LOCATE 17, 34: COLOR 0, C(5): PRINT " 5 "; : COLOR CI(5), 0: PRINT "  FORMAT"
   LOCATE 20, 34: COLOR 0, C(0): PRINT " 0 "; : COLOR CI(0), 0: PRINT "  MAIN MENU"
   Menu 5: IF K = 0 THEN GOTO MainMenu
   IF K = 5 THEN Text$ = "  RETURN TO NAV DATA MENU": LatLngFormat: GOTO NavDataEntry
   IF K = 2 THEN GOTO DeleteNavData ELSE IF K > 2 THEN Printer = K - 3: GOTO ListNavData

'Add/Edit the Nav File
   CALL NavFile: Hue = 1: Heading$ = "ADD/EDIT NAV DATA": ScreenHeading: FormatDisplay
NavAddEdit:
   COLOR CI(Hue): LOCATE 5, 21: PRINT NRec; "Entries   ": AiportNavaid
   DataAdded = No: IF K THEN GOSUB SortNavData: GOTO NavDataEntry
   Changed = No: Escape "entry": LOCATE 5, 22: PRINT Fac$; "      "
   Edit = No: Editing = No: GOSUB Identifier: IF Max = 1 THEN GOTO EndAddEdit
   IF Ndx THEN
      RecNdx = Ndx + 1: NavRec: Navaid
      IF AF$ = "X" THEN
         COLOR CI(0) + 16: LOCATE 5, 22: PRINT "DELETED": COLOR CI(Hue): Continue 21: GOTO EndAddEdit
      END IF
      Editing = Yes: PageFlag = Yes: Escape "change": GOTO EditFacData
   END IF
   Rec = NRec
   IF NRec > 999 THEN 'Full Nav Data File
      LOCATE 17, 30: COLOR C(0), 0: PRINT "NAV DATA FILE IS FULL!"
      PRINT CHR$(7): Time 2: GOSUB SortNavData: GOTO NavDataEntry
   END IF

AddFacData:
   LOCATE 5, 22: PRINT "ADD "; Fac$; " ": PageFlag = No
   FOR I = 1 TO 4
      ON I GOSUB Latitude, Longtitude, Variation, RadioFreq: IF Max = 1 THEN GOTO EndAddEdit
```

```
   NEXT: DataAdded = Yes

EditFacData:
   LOCATE 5, 22: PRINT "EDIT "; Fac$; " "
   DO
      FieldChoice 5, 21, PageFlag: IF K$ = Rtn$ THEN Max = 2: GOTO EndAddEdit
      IF K$ = Esc$ THEN DataAdded = No: Max = 1: GOTO EndAddEdit
      IF K$ = "-" OR K$ = "+" THEN
         GOSUB StoreNavdata
         DO
            Index = Index - ASC(K$) + 44
            Index = -Index * (Index > 0 AND Index < NRec) - 0 * (Index > NRec) - (NRec - 1) * (Index < 0)
            Rec = VAL(MID$(Index$, Index * 9 + 6, 3)): GET 1, Rec + 2: RecNdx = Index * 9 + 2
         LOOP WHILE AF$ = "X": Id = VAL(AF$): Facility: Navaid: GOTO EditFacData
      ELSE
         Wipe 21, 1: LOCATE 25, 8: PRINT SPC(18); : LOCATE 25, 57: PRINT SPC(14); : Edit = Yes
         ON VAL(K$) GOSUB Edit1, Edit2, Edit3, Edit4, Edit5
      END IF
   LOOP
Edit1: Text$ = FormatId(Id$): GOTO GetId
Edit2: Text$ = STR$(VAL(Lat$)): Strip: GOTO GetLat
Edit3: Text$ = STR$(VAL(Lng$)): Strip: GOTO GetLng
Edit4: Text$ = Var$: MID$(Text$, 1, 1) = CHR$(87 + 18 * (VAL(Text$) < 0)): GOTO GetVar
Edit5: Text$ = Freq$: GOTO GetRadioFreq

EndAddEdit:
   IF Routing THEN RETURN ELSE GOSUB StoreNavdata: GOSUB Clean: GOTO NavAddEdit

Clean:
   Wipe 9, 1: Wipe 11, 1: Wipe 13, 1: Wipe 15, 1: Wipe 17, 1: Wipe 21, 1: RETURN

Identifier:
   LOCATE 9, 26: COLOR 0, C(Hue): PRINT " 1 "; : COLOR CI(Hue), 0: PRINT "  Identifier:"
GetId:
   Wipe 9, 43: Query 9, 43, 1 - Edit * 10, 3 + Id + (Id = 3): Strip
   IF Max > 1 THEN
      IF LEN(Text$) = 0 THEN
         IF Edit THEN Text$ = "* INVALID *": GOTO IdError
         Max = 1: RETURN
      END IF: IF Id = 3 THEN RETURN
      IF Id = 0 OR LEN(Text$) < 4 THEN Text$ = " " + LEFT$(Text$ + "   ", 3)
      Ndx$ = Text$: Ndx = INSTR(Index$, AF$ + Ndx$)
      IF Ndx = 0 THEN
         IF Deleting THEN Text$ = "* NOT FOUND *": GOTO IdError
         LSET Id$ = Text$
         IF Editing THEN DataAdded = Yes: Changed = Yes: MID$(Index$, RecNdx, 4) = Ndx$
      ELSE
        IF NOT Edit THEN RETURN ELSE IF Ndx$ <> Id$ THEN Text$ = "* DUPLICATE *": GOTO IdError
      END IF
   ELSE
      IF NOT Edit THEN RETURN
   END IF: LOCATE 9, 43: PRINT Id$: RETURN
IdError:
   Wipe 9, 43: LOCATE 9, 43: COLOR 0, C(0): PRINT Text$: BEEP: Time 1: Wipe 9, 43: GOTO Edit1

Latitude:
   LOCATE 11, 26: COLOR 0, C(Hue): PRINT " 2 "; : COLOR CI(Hue), 0: PRINT "  Latitude:"
GetLat:
   Wipe 11, 41: Query 11, 41, 3 - Edit * 10, 8 - F
   IF Max > 1 THEN
      Sign = SGN(VAL(Text$)): Text$ = STR$(ABS(VAL(Text$)) * 10000 + 1000000!)
      Text$ = MID$(Text$, 3, 2) + "." + MID$(Text$, 5, 4 - F): Length = 1
      IF ASC(Text$) = 48 THEN MID$(Text$, 1, 1) = " ": Length = 2
      Text$ = " " + Text$: MID$(Text$, Length, 1) = CHR$(45 + (Sign > -1) * 13)
      IF ABS(VAL(Text$)) > 90 OR VAL(MID$(Text$, 5, 2)) > 59 OR (F = 0 AND VAL(MID$(Text$, 7, 2)) > 59) THEN
         Wipe 11, 41: LOCATE 11, 41: COLOR 0, C(0): PRINT "* INVALID *"
         BEEP: Time 1: Wipe 11, 41: COLOR CI(Hue): GOTO Edit2
      END IF: LSET Lat$ = Text$: LatFormat: IF Edit THEN DataAdded = Yes
   ELSE
      IF Edit THEN LatFormat ELSE RETURN
   END IF: LOCATE 11, 41: PRINT Deg$: RETURN

Longtitude:
   LOCATE 13, 26: COLOR 0, C(Hue): PRINT " 3 "; : COLOR CI(Hue), 0: PRINT "  Longitude:"
GetLng:
   Wipe 13, 42: Query 13, 42, 3 - Edit * 10, 9 - F
```

```
   IF Max > 1 THEN
      Sign = SGN(VAL(Text$)): Text$ = STR$(ABS(VAL(Text$)) * 10000 + 10000000#)
      Text$ = MID$(Text$, 3, 3) + "." + MID$(Text$, 6, 4 - F): Length = 1
      IF ASC(Text$) = 48 THEN
         MID$(Text$, 1, 1) = " ": Length = 2
         IF MID$(Text$, 2, 1) = "0" THEN MID$(Text$, 2, 1) = " ": Length = 3
      END IF
      Text$ = " " + Text$: MID$(Text$, Length, 1) = CHR$(45 + (Sign > -1) * 13)
      IF ABS(VAL(Text$)) > 180 OR VAL(MID$(Text$, 6, 2)) > 59 OR (F = 0 AND VAL(MID$(Text$, 8, 2)) > 59) THEN
         Wipe 13, 42: LOCATE 13, 42: COLOR 0, C(0): PRINT "* INVALID *"
         BEEP: Time 1: Wipe 13, 42: COLOR CI(Hue), 0: GOTO Edit3
      END IF: LSET Lng$ = Text$: LngFormat: IF Edit THEN DataAdded = Yes
   ELSE
      IF Edit THEN LngFormat ELSE RETURN
   END IF: LOCATE 13, 42: PRINT Deg$: RETURN

Variation:
   LOCATE 15, 26: COLOR 0, C(Hue): PRINT " 4 "; : COLOR CI(Hue), 0: PRINT "  Magnetic Variation:"
GetVar:
   Wipe 15, 51: Query 15, 51, 4 - Edit * 10, 3
   IF Max > 1 THEN
      IF ASC(Text$) > 47 AND ASC(Text$) < 59 THEN Var = VAL(Text$) ELSE Var = VAL(RIGHT$(Text$, 2))
      IF INSTR(Text$, "E") OR INSTR(Text$, "-") THEN Var = -Var
      LSET Var$ = STR$(Var): VarFormat: IF Edit THEN DataAdded = Yes
   ELSE
      IF Edit THEN VarFormat ELSE RETURN
   END IF: LOCATE 15, 51: PRINT Deg$: RETURN

RadioFreq:
   LOCATE 17, 26: COLOR 0, C(Hue): PRINT " 5 "; : COLOR CI(Hue), 0: PRINT "  Radio Frequency:"
GetRadioFreq:
   Query 17, 48, 0 - Edit * 10, 7: IF K$ = Esc$ AND NOT Edit THEN RETURN
   IF K$ <> Esc$ THEN LSET Freq$ = Text$: DataAdded = Yes
   Max = 2: LOCATE 17, 48: PRINT Freq$: RETURN

StoreNavdata:
   IF DataAdded THEN
      PUT 1, Rec + 2: IF Changed THEN Sort = Yes
      IF NOT Editing THEN
         Index$ = Index$ + AF$ + Ndx$ + RIGHT$(STR$(Rec + 1000), 3) + CHR$(254)
         GET 1, 1: NRec = Rec + 1: LSET Rec$ = MKI$(NRec): PUT 1, 1: GET 1, Rec + 2: Sort = Yes
      END IF
   END IF: RETURN

SortNavData:
   IF Sort THEN
      Heading$ = "SORTING THE NAV DATA": ScreenHeading: COLOR CI(4): LOCATE 12, 34: PRINT "Records sorted"
      COLOR C(Hue), 0: LOCATE 16, 2: PRINT "1 "; : COLOR 0, C(Hue): PRINT STRING$(76 - LEN(STR$(NRec)), 8);
      COLOR C(Hue), 0: PRINT STR$(NRec): Escape "sorting": Length = 76 - LEN(STR$(NRec)): COLOR CI(4), C(Hue)
      Begin = LEN(Index$) - 17: Limit = 1: Sorted = No: SortedPortion# = Length / (Begin + 9)
      DO
         Sorting = No: LOCATE 16, 4: PRINT STRING$(Limit * SortedPortion#, 8)
         FOR X = Begin TO Limit STEP -9: Y = X + 9
            IF ASC(MID$(Index$, X + 2, 1)) > 64 AND ASC(MID$(Index$, Y + 2, 1)) < 65 THEN GOTO SkipSort
            IF ASC(MID$(Index$, X + 2, 1)) < 65 AND ASC(MID$(Index$, Y + 2, 1)) > 64 THEN GOTO DoSort
            IF MID$(Index$, X + 2, 3) = MID$(Index$, Y + 2, 3) THEN
               IF MID$(Index$, X, 2) > MID$(Index$, Y, 2) THEN GOTO DoSort ELSE GOTO SkipSort
            END IF
            IF MID$(Index$, X + 2, 3) > MID$(Index$, Y + 2, 3) THEN
DoSort:
               Temp$ = MID$(Index$, X, 8): MID$(Index$, X, 8) = MID$(Index$, Y, 8)
               MID$(Index$, Y, 8) = Temp$: Sorting = Y: Sorted = Yes
            END IF
SkipSort:
            IF INKEY$ = Esc$ THEN Sorting = No: Sorted = No: EXIT FOR
         NEXT: Limit = Sorting
      LOOP WHILE Sorting: LOCATE 16, 4: PRINT STRING$(Length, 8)
      OPEN "NavFile.NDX" FOR OUTPUT AS 2: PRINT #2, Index$
   END IF: CLOSE : Sort = No: RETURN

DeleteNavData:
   CALL NavFile: Hue = 2: Heading$ = "DELETE NAV DATA": ScreenHeading
   Deletions = No: Edit = No: IF NRec = 0 THEN GOTO EmptyFile
   DO
      Wipe 5, 22: COLOR CI(0) + 16: LOCATE 5, 22: PRINT "DELETE": COLOR CI(Hue)
      LOCATE 5, 52 - LEN(STR$(NRec - Deleted)): PRINT NRec - Deleted; "Entries": AiportNavaid
```

```
      IF K THEN 'Pack & Reindex
         IF Deletions THEN
            Heading$ = "PACK NAV DATA": ScreenHeading: Escape "packing": Deletions = No
            LOCATE 14, 36: PRINT "Packing": Temp$ = "": NewRec = 0: TempNavFile
            FOR Rec = 0 TO NRec - 1
               GET 1, Rec + 2
               IF AF$ <> "X" THEN
                  LSET Dat$ = Old$: PUT 2, NewRec + 2: Temp$ = Temp$ + AF$ + Id$
                  Temp$ = Temp$ + RIGHT$("00" + RIGHT$(STR$(NewRec), LEN(STR$(NewRec)) - 1), 3) + CHR$(254)
                  NewRec = NewRec + 1: LOCATE 14, 43: PRINT NewRec
               END IF: IF INKEY$ = Esc$ THEN CLOSE : KILL "TempNav.dat": GOTO NavDataEntry
            NEXT: GET 2, 1: LSET Dat$ = MKI$(NewRec): PUT 2, 1: CLOSE
            KILL "NavFile.DAT": NAME "TempNav.DAT" AS "NavFile.DAT": Deleted = 0
            Index$ = Temp$: OPEN "NavFile.NDX" FOR OUTPUT AS 2: PRINT #2, Index$
         END IF: CLOSE : GOTO NavDataEntry
      END IF: Escape "": LOCATE 5, 29: PRINT Fac$: Deleting = Yes: GOSUB Identifier: Deleting = No
      IF Max > 1 THEN
         CALL NavRec: Navaid: Text$ = FormatId(Id$): Strip
         LOCATE 21, 18: PRINT "Are you sure you want to "; : COLOR CI(0): PRINT "delete ";
         COLOR CI(Hue): PRINT Text$; : PRINT " (Y/N)? "; : COLOR C(Hue): PRINT CHR$(222);
         COLOR 16, C(Hue): PRINT "N"; : COLOR C(Hue), 0: PRINT CHR$(221): COLOR CI(Hue)
         DO
            K$ = UCASE$(INPUT$(1)): IF K$ = Esc$ THEN EXIT DO
            IF K$ = "Y" THEN
               Deletions = Yes: IF AF$ <> "X" THEN Deleted = Deleted + 1: LSET AF$ = "X"
            ELSEIF K$ = Rtn$ OR K$ = "N" THEN
               K$ = "Y": IF AF$ = "X" THEN LSET AF$ = RIGHT$(STR$(Id), 1): Deleted = Deleted - 1
               IF Deleted < 0 THEN Deleted = 0
            END IF: IF K$ = "Y" THEN PUT 1, Rec + 2
         LOOP WHILE K$ <> "Y"
      END IF: GOSUB Clean
   LOOP

ListNavData:
   CALL NavFile: Hue = Printer + 3: Msg$ = MID$("LISTPRINT", 1 + 4 * Printer, 4 + Printer)
   Heading$ = Msg$ + " NAV DATA": ScreenHeading: IF NRec = 0 THEN GOTO EmptyFile
   Text$ = STR$(NRec) + " Entries ": Col = 1 + INT(80 - LEN(Text$)) / 2: LOCATE 6, Col: PRINT Text$
   IF Printer THEN LOCATE 9, 21: PRINT "Prepare Printer {top-of-form & on-line}"
   LOCATE 11 + Printer * 2, 30: COLOR 0, C(Hue): PRINT " RETURN "; : COLOR CI(Hue), 0: PRINT "  All Nav Data"
   LOCATE 13 + Printer * 2, 35: COLOR 0, C(Hue): PRINT " 1 "; : COLOR CI(Hue), 0: PRINT "  Navaids"
   LOCATE 15 + Printer * 2, 35: COLOR 0, C(Hue): PRINT " 2 "; : COLOR CI(Hue), 0: PRINT "  Airports": Escape ""
   LOCATE 20 + Printer, 30: PRINT Msg$; " which Nav Data? "; : COLOR CI(Hue) + 16: PRINT CHR$(22)
   DO
      K$ = INPUT$(1): IF K$ = Esc$ THEN CLOSE : GOTO NavDataEntry
   LOOP WHILE K$ <> Rtn$ AND K$ <> "1" AND K$ <> "2"
   Id = VAL(K$) - 1: AF = Id: IF Id < 0 THEN Fac$ = "NAV" ELSE Facility
   Top$ = Fac$ + " DATA": Heading$ = Msg$ + " " + Top$: Rec = 0: Lines = 15 + Printer * 35
   IF Printer THEN
      Pages = INT(NRec / 50 + .99): Degree$ = PrDeg$: Page = 1: ScreenHeading
      LOCATE 14, 37: COLOR CI(Hue) + 16: PRINT "PRINTING": Escape "printing"
   END IF: Hdg$ = "Facility         Latitude        Longitude       Variation    Frequency"
OutputHeading:
   IF Printer THEN
      ON ERROR GOTO NavDataPrintError: LPRINT PrInit$; SPACE$((80 - LEN(Top$)) / 2); Top$
      LPRINT TAB(4); "Printed: "; MID$("  JanFebMarAprMayJunJulAugSepOctNovDec", VAL(LEFT$(DATE$, 2)) * 3, 3);
      LPRINT " "; MID$(DATE$, 4, 2); ", "; RIGHT$(DATE$, 4); TAB(71 - LEN(STR$(Pages)) - LEN(STR$(Page)));
      LPRINT SPACE$((-3 - LEN(STR$(Pages))) * (AF > -1)); "Page"; Page; : IF AF < 0 THEN LPRINT "of"; Pages
      LPRINT TAB(4); STRING$(74, DoubleLine$): LPRINT TAB(7); Hdg$
      LPRINT TAB(4); STRING$(74, SingleLine$): LPRINT
   ELSE
      CALL ScreenHeading: LOCATE 4, 7: PRINT Hdg$
      LOCATE 5, 4: PRINT STRING$(74, 196): Escape "listing": LOCATE 7, 1
   END IF: LineNo = 0
LineOutput:
   IF INKEY$ = Esc$ OR Rec = NRec THEN GOTO QuitListing
   GET 1, VAL(MID$(Index$, Rec * 9 + 6, 3)) + 2: Rec = Rec + 1: Id = VAL(AF$) + (AF$ = "X")
   IF AF < 0 OR AF = Id THEN Facility ELSE GOTO LineOutput
   Text$ = SPACE$(-5 * (AF > -1) + (AF = 0)) + Id$: IF AF < 0 THEN Text$ = Text$ + " - " + Fac$
   Text$ = Text$ + SPACE$(4 - 5 * (AF > -1) + Format - (Id = 0)): LatFormat
   Text$ = Text$ + Deg$ + SPACE$(4 + 2 * Format): LngFormat: LineNo = LineNo + 1
   Text$ = Text$ + Deg$ + SPACE$(7 + Format): VarFormat: Text$ = Text$ + Deg$ + "        " + Freq$
   IF Printer THEN LPRINT TAB(4); Text$ ELSE PRINT TAB(4); Text$
   IF LineNo < Lines THEN GOTO LineOutput
   IF Rec < NRec THEN
      IF Printer THEN LPRINT CHR$(12);  ELSE Continue 23: IF K$ = Esc$ THEN CLOSE : GOTO NavDataEntry
      Page = Page + 1: GOTO OutputHeading
```

```
   END IF
QuitListing:
   IF Printer THEN LPRINT CHR$(12); : Degree$ = DegreeSymbol$ ELSE Continue 23
   CLOSE : ON ERROR GOTO 0: GOTO NavDataEntry

FlightLoad:
   Heading$ = "LOAD FLIGHT DATA": ScreenHeading: Escape "": FlightIndex
   IF K$ = Esc$ THEN GOTO MainMenu
LoadMenu:
   LOCATE 22, 31: COLOR 7, 0: PRINT "Load which flight? "; : COLOR 23: PRINT CHR$(22)
   IF K$ = "" OR K$ < "A" OR K$ > "Z" THEN
      DO
         K$ = UCASE$(INPUT$(1)): IF K$ = Esc$ OR K$ = Rtn$ THEN GOTO MainMenu
      LOOP WHILE K$ < "A" OR K$ > "Z"
   END IF: OPEN "FltData." + K$ FOR INPUT AS 2: INPUT #2, Id1$
   IF Id1$ = "EMPTY" THEN
      LOCATE 22, 50: COLOR 0, C(0): PRINT "* EMPTY *": Time 1
      CLOSE 2: K$ = "": COLOR 7, 0: Wipe 22, 49: GOTO LoadMenu
   END IF: ON ERROR GOTO 0
   INPUT #2, Id2$, Flight$: FlightDisplay: CLOSE
   IF K$ = Esc$ THEN GOTO MainMenu ELSE IF K$ = "N" THEN GOTO FlightLoad ELSE GOTO RouteMenu

FlightSave:
   Id1$ = Departure$: Id2$ = Destination$: IF Direction < 0 THEN SWAP Id1$, Id2$
   Heading$ = "SAVE FLIGHT " + Id1$ + " to " + Id2$: ScreenHeading
   Escape "": FlightIndex: ON ERROR GOTO 0: IF K$ = Esc$ THEN GOTO RouteMenu
   LOCATE 22, 28: COLOR 7, 0: PRINT "Store in which position? "; : COLOR 23: PRINT CHR$(22)
   IF K$ = "" OR K$ < "A" OR K$ > "Z" THEN
      DO
         K$ = UCASE$(INPUT$(1)): IF K$ = Esc$ OR K$ = Rtn$ THEN GOTO RouteMenu
      LOOP WHILE K$ < "A" OR K$ > "Z"
   END IF: COLOR 7, 0: LOCATE 22, 53: PRINT K$; : OPEN "FltData." + K$ FOR OUTPUT AS 2
   WRITE #2, Id1$, Id2$, Flight$: CLOSE : GOTO RouteMenu

PrinterCodes:
   Heading$ = "PRINTER CODES": ScreenHeading: LOCATE 6, 16: COLOR 0, 7
   PRINT " 1 "; : COLOR 7, 0: PRINT " Printer Initialization:": LOCATE 8, 19
   FOR I = 1 TO 24
      IF I = 13 THEN LOCATE 10, 19
      PRINT Zero(MID$(PrInit$, I, 1)); " ";
   NEXT: LOCATE 13, 16: COLOR 0, 7: PRINT " 2 "; : COLOR 7, 0: PRINT " Degree Symbol:": LOCATE 15, 19
   FOR I = 1 TO 24
      IF I = 13 THEN LOCATE 17, 19
      PRINT Zero(MID$(PrDeg$, I, 1)); " ";
   NEXT: LOCATE 20, 16: COLOR 0, 7: PRINT " 3 "; : COLOR 7, 0
   PRINT " Single Line ("; CHR$(196); "): "; : PRINT Zero(SingleLine$)
   LOCATE 20, 46: PRINT "Double Line ("; CHR$(205); "): "; : PRINT Zero(DoubleLine$): Escape ""

EnterPrCodes:
   Hue = 1: FieldChoice 3, 23, 0: Hue = 6: Wipe 23, 18: IF K = 0 THEN GOTO MainMenu
   LOCATE 23, 29: PRINT "ENTER DECIMAL ASCII CODES": COLOR 7, 0
   IF K = 3 THEN
      Text$ = "": K$ = STR$(ASC(SingleLine$)): IF VAL(K$) THEN Text$ = RIGHT$(K$, LEN(K$) - 1)
      Query 20, 37, 12, 3: IF K$ = Esc$ THEN GOTO PrinterCodes
      K$ = CHR$(VAL(Text$) MOD 256): LOCATE 20, 37: PRINT Zero(K$): SingleLine$ = K$
      Text$ = "": K$ = STR$(ASC(DoubleLine$)): IF VAL(K$) THEN Text$ = RIGHT$(K$, LEN(K$) - 1)
      Query 20, 63, 12, 3: IF K$ = Esc$ THEN GOTO PrinterCodes
      K$ = CHR$(VAL(Text$) MOD 256): LOCATE 20, 63: PRINT Zero(K$): DoubleLine$ = K$
   ELSE
      IF K = 1 THEN PR$ = PrInit$ ELSE PR$ = PrDeg$
      FOR I = 1 TO 24
         Row = 8 + (K - 1) * 7 - (I > 12) * 2: Col = 15 + I * 4 + (I > 12) * 48
         Text$ = "": K$ = STR$(ASC(MID$(PR$, I, 1)))
         IF VAL(K$) THEN Text$ = RIGHT$(K$, LEN(K$) - 1)
         Query Row, Col, 12, 3: IF K$ = Esc$ THEN GOTO PrinterCodes
         K$ = CHR$(VAL(Text$) MOD 256): LOCATE Row, Col
         PRINT Zero(K$): MID$(PR$, I, 1) = K$: IF Max = 1 THEN I = 24
      NEXT
   END IF: SysFile
   IF K = 1 THEN PrInit$ = PR$: LSET Pr1$ = PR$
   IF K = 2 THEN PrDeg$ = PR$: LSET Pr2$ = PR$
   IF K = 3 THEN LSET SL$ = SingleLine$: LSET DL$ = DoubleLine$
   PUT 3, 1: CLOSE 3: GOTO EnterPrCodes

ScreenInit:
   Heading$ = "SCREEN INITIALIZATION": ScreenHeading
```

```
      LOCATE 11, 30: COLOR 0, 7: PRINT " 1 "; : COLOR 7, 0: PRINT "  MONOCHROME"
   LOCATE 13, 30: COLOR 0, 7: PRINT " 2 "; : COLOR 7, 0: PRINT "  COLOR"
   LOCATE 21, 26: PRINT "Which monitor do you have? "; : COLOR 23: PRINT CHR$(22): COLOR 7, 0
   DO
      K$ = INPUT$(1): IF K$ = Rtn$ OR K$ = Esc$ THEN GOTO MainMenu
   LOOP WHILE K$ < "1" OR K$ > "2": LOCATE 21, 53: PRINT K$
   CALL SysFile: CFlg = VAL(K$) - 1: LSET CFlg$ = RIGHT$(STR$(CFlg), 1)
   PUT 3, 1: CLOSE : C$ = "777777": IF CFlg THEN C$ = "465312"
   FOR I = 0 TO 5: C(I) = VAL(MID$(C$, I + 1, 1)): IF CFlg THEN CI(I) = C(I) + 8 ELSE CI(I) = C(I)
   NEXT: GOTO MainMenu

EmptyFile:
   COLOR CI(Hue), 0: LOCATE 14, 29: PRINT "Nav Data File is empty!": CLOSE : Continue 24

FlightFileError:
   IF ERR = 53 THEN OPEN "FltData." + K$ FOR OUTPUT AS 2: WRITE #2, "EMPTY": CLOSE : RESUME
   ON ERROR GOTO 0

IndexFileError:
   IF ERR = 53 THEN OPEN "NavFile.NDX" FOR OUTPUT AS 2: PRINT #2, "": CLOSE : RESUME
   ON ERROR GOTO 0

NavDataPrintError:
   GOSUB PrinterError: RESUME NavDataEntry

RoutePrintError:
   GOSUB PrinterError: RESUME RouteMenu

PrinterError:
   IF ERR = 24 OR ERR = 25 THEN
      CALL ScreenHeading: COLOR C(0) + 15: LOCATE 14, 34: PRINT "PRINTER ERROR!"; CHR$(7)
      CLOSE : COLOR C(0): Continue 24: RETURN
   END IF: ON ERROR GOTO 0


'******* SUBROUTINES *******


FUNCTION ACos# (X#) STATIC
   IF X# = -1 THEN ACos# = PI#: EXIT FUNCTION
   ACos# = D90# - ASin(X#)
END FUNCTION

DEFINT A-Z
SUB AiportNavaid STATIC
   LOCATE 11, 35: COLOR 0, C(Hue): PRINT " 1 "; : COLOR CI(Hue), 0: PRINT "  Navaid"
   LOCATE 13, 35: COLOR 0, C(Hue): PRINT " 2 "; : COLOR CI(Hue), 0: PRINT "  Airport"
   LOCATE 17, 32: COLOR CI(Hue), 0: PRINT "Navaid or Airport? ";
   COLOR CI(Hue) + 16: PRINT CHR$(22): Escape ""
   DO
      K$ = INPUT$(1): : IF K$ = Esc$ OR K$ = Rtn$ THEN K = 1: EXIT SUB
   LOOP WHILE K$ <> "1" AND K$ <> "2": Id = VAL(K$) - 1: K = 0
   Wipe 11, 35: Wipe 13, 35: Wipe 17, 32: LSET AF$ = RIGHT$(STR$(Id), 1): Facility
END SUB

FUNCTION ASin# (X#) STATIC
   IF ABS(X#) = 1 THEN ASin# = PI# - SGN(X#) * D90#: EXIT FUNCTION
   ASin# = ATN(X# / SQR(ABS(1 - X# * X#)))
END FUNCTION

DEFINT A-Z
SUB Continue (Row) STATIC
   COLOR CI(Hue): LOCATE Row, 28: PRINT "Press any key to continue ";
   COLOR CI(Hue) + 16: PRINT CHR$(22); : COLOR CI(Hue): K$ = INPUT$(1): Wipe Row, 1: LOCATE 1, 1
END SUB

DEFINT A-Z
SUB Escape (Msg$) STATIC
   COLOR 0, 0: Wipe 25, 1: EscCol = 43 - CINT(LEN(Msg$) / 2): LOCATE 25, EscCol - 9
   COLOR 0, C(0): PRINT " ESC "; : COLOR CI(0), 0: PRINT " to abort " + Msg$; : COLOR CI(Hue), 0
END SUB

DEFINT A-Z
SUB Facility STATIC
   Fac$ = "Navaid": IF Id THEN Fac$ = "Airport": IF Id < 1 THEN Fac$ = "DELETED"
   IF Id > 1 THEN Fac$ = "Waypoint": IF Id = 3 THEN Fac$ = "Intersection"
```

```
'NOTES:
   '0 = NAVAID
   '1 = AIRPORT
   '2 = WAYPOINT
   '3 = INTERSECTION
   '4 = TERMINAL WAYPOINT
   '5 = END OF FLIGHT
END SUB

DEFINT A-Z
SUB FieldChoice (UpperLimit, Row, PageFlag) STATIC
   LOCATE Row, 18: COLOR CI(Hue), 0: PRINT "Enter Field to Change - or ";
   COLOR 0, C(Hue): PRINT " RETURN "; : COLOR CI(Hue), 0: PRINT " to Accept ";
   COLOR CI(Hue) + 16: PRINT CHR$(22); : K$ = ""
   IF PageFlag THEN
      LOCATE 25, 8: COLOR 0, C(Hue): PRINT " - "; : COLOR CI(Hue), 0: PRINT " Previous Entry";
      LOCATE 25, 57: COLOR 0, C(Hue): PRINT " + "; : COLOR CI(Hue), 0: PRINT " Next Entry";
   END IF: COLOR CI(Hue)
   WHILE K$ < "1" OR K$ > RIGHT$(STR$(UpperLimit), 1)
      K$ = INPUT$(1): K = VAL(K$): IF K$ = Esc$ OR K$ = Rtn$ THEN EXIT SUB
      IF PageFlag AND (K$ = "-" OR K$ = "+") THEN EXIT SUB
   WEND
END SUB

DEFINT A-Z
SUB FileCheck
   IF NRec = LEN(Index$) / 9 THEN EXIT SUB
   Heading$ = "FILE-INDEX ERROR": Hue = 0: ScreenHeading
   LOCATE 13, 26: PRINT "The NAV DATA file needs to be"
   LOCATE 15, 35: PRINT "Re-indexed!": Escape ""
   Continue 20: IF K$ = Esc$ THEN COLOR 7, 0: CLS : END ELSE IndexFile
END SUB

DEFINT A-Z
SUB FlightDisplay STATIC
   Heading$ = "FLIGHT ROUTE": Hue = 2: ScreenHeading: Escape "listing": Row = 7: Pointer = 1: Departure = Yes
ListNav:
   Id = VAL(MID$(Flight$, Pointer, 1)): Facility
   IF Id < 4 THEN
      K$ = INKEY$: IF K$ = Esc$ THEN EXIT SUB
      LOCATE Row, 26: Row = Row + 1
      IF Row = 23 THEN
         Msg$ = "Partial listing from " + Departure$: LOCATE 5, 28: PRINT Msg$
         Continue 23: IF K$ = Esc$ THEN EXIT SUB
         CALL ScreenHeading: LOCATE 5, 28: PRINT Msg$: Escape "listing"
         LOCATE 7, 26: PRINT LastFix$: Row = 9: LOCATE 8, 26
      END IF
   END IF
   ON Id GOTO Id1, Id1, Id3, Id4, Id5
Id1:
   Text$ = MID$(Flight$, Pointer + 1, 4): Strip: LastFix$ = " " + RIGHT$("    " + Text$, 4) + " " + Fac$
   PRINT LastFix$; : Pointer = Increment(Pointer, VAL(MID$(Flight$, Pointer + 5, 1))): GOTO DepartCk
Id3:
   Text$ = MID$(Flight$, Pointer + 1, 5): Strip: PRINT RIGHT$("     " + Text$, 5) + " " + Fac$
   Pointer = Increment(Pointer, VAL(MID$(Flight$, Pointer + 10, 1))) + 15
DepartCk:
   IF Departure THEN Departure$ = Text$: Departure = No
   GOTO ListNav
Id4:
   PRINT " * " + Fac$ + " " + MID$(Flight$, Pointer + 2, 3)
   Pointer = Increment(Pointer, VAL(MID$(Flight$, Pointer + 5, 1))): GOTO ListNav
Id5:
   Destination$ = Text$: Heading$ = Departure$ + " to " + Destination$
   Wipe 5, 28: LOCATE 5, 41 - LEN(Heading$) / 2: PRINT Heading$: Direction = 1
   LOCATE 23, 28: COLOR 7, 0: PRINT "Is this correct (Y/N)? "; CHR$(222);
   COLOR 16, 7: PRINT "Y"; : COLOR 7, 0: PRINT CHR$(221); : Escape "routing"
   DO
      K$ = UCASE$(INPUT$(1))
   LOOP WHILE K$ <> Esc$ AND K$ <> "N" AND K$ <> "Y" AND K$ <> Rtn$
END SUB

DEFINT A-Z
SUB FlightIndex STATIC
   ON ERROR GOTO FlightFileError
   FOR I = 0 TO 1: Col = I * 22 + 22: FOR Row = 7 TO 19: K$ = CHR$(13 * I + Row + 58)
      OPEN "FltData." + K$ FOR INPUT AS 2: INPUT #2, Text$
```

```
      IF Text$ <> "EMPTY" THEN INPUT #2, Temp$: Text$ = Text$ + " to " + Temp$
      CLOSE : LOCATE Row, Col: COLOR 0, C(2): PRINT CHR$(221) + K$ + CHR$(222);
      COLOR CI(2), 0: PRINT " "; Text$
      K$ = UCASE$(INKEY$): IF K$ = Esc$ THEN I = 1: Row = 19
      IF (K$ > "@" AND K$ < "[") THEN EXIT SUB
   NEXT: NEXT: ON ERROR GOTO 0
END SUB

DEFINT A-Z
SUB FormatDisplay STATIC
      A$ = "Format: DD" + DegreeSymbol$ + " MM"
      IF Format THEN A$ = A$ + ".d'" ELSE A$ = A$ + "' SS" + CHR$(34)
      LOCATE 5, (41 + Format * 2): PRINT A$
END SUB

FUNCTION FormatId$ (A$) STATIC
   Text$ = A$: Strip: FormatId$ = Text$
END FUNCTION

DEFINT A-Z
FUNCTION Increment (Pointer, CoordinateFlag) STATIC
   Increment = Pointer + 6 + CoordinateFlag * 27
'NOTES: Flight$ (Flight route data)
   'Navaid, Airport, or Waypoint -
      'Byte 1:                    Fix identifier (see Sub Facility)
      'Bytes 2 to 5:              Four letter identifier
      'Byte 6:                    Coordinate Flag (0 = in NavFile, 1 = in Flight$)
      'Bytes 7 to 33:             Coordinate data if stored in the Flight$
      '  Bytes 7 to 14:              Latitude (8 bytes - bytes 1 to 8 relative)
      '  Bytes 15 to 23:             Longtitude (9 bytes - bytes 9 to 17 relative)
      '  Bytes 24 to 26:             Variation (3 bytes - bytes 18 to 20 relative)
      '  Bytes 27 to 33:             Frequence (7 bytes - bytes 21 to 27 relative)
   'Intersection -
      'Byte 1:                    "3" for intersection
      'Bytes 2 to 6:              Five letter identifier
      'Bytes 7 to 11(38):         Reference Navaid Data
      '  Bytes 7 to 10:              Four letter identifier
      '  Byte 11:                    Coordinate Flag
      '  Bytes (12) to (38):         Coordinate data if stored in the Flight$
      'Bytes 12(39) to 16(43):    Radial (5 bytes - 000.1 to 360.0)
      'Bytes 17(44) to 21(48):    DME Distance (5 bytes - 0.0 to 999.9)
END FUNCTION

DEFINT A-Z
SUB IndexFile STATIC
   Hue = 5: ScreenHeading: LOCATE 14, 35: PRINT "Indexing": Rec = 0: Index$ = ""
   DO
      GET 1, Rec + 2
         IF ASC(Rec$) = 0 THEN
         EXIT DO
      ELSE
         Index$ = Index$ + AF$ + Id$ + RIGHT$("00" + RIGHT$(STR$(Rec), LEN(STR$(Rec)) - 1), 3) + CHR$(254)
         Rec = Rec + 1: LOCATE 14, 43: PRINT Rec
      END IF: GET 1, 1: LSET Rec$ = MKI$(Rec): PUT 1, 1: NRec = Rec
   LOOP: OPEN "NavFile.NDX" FOR OUTPUT AS 2: PRINT #2, Index$: CLOSE 2
END SUB

DEFINT A-Z
SUB IntroScreen STATIC
   COLOR C(4), C(4): CLS : COLOR 15: LOCATE 2, 40: PRINT CHR$(15)
   LOCATE 3, 37: COLOR CI(3): PRINT "IMKOKM;"
   LOCATE 4, 37: PRINT ": : : :"
   LOCATE 5, 18: COLOR 8: PRINT ".": LOCATE 5, 37: COLOR CI(3): PRINT "HKJMJK<"
   LOCATE 6, 16: COLOR 8: PRINT "DMmMD": LOCATE 6, 38: COLOR CI(3): PRINT ":   :"
   LOCATE 7, 38: PRINT ":   :"; : LOCATE 7, 57: COLOR CI(0): PRINT "^[[["; CHR$(16)
   LOCATE 8, 38: COLOR CI(3): PRINT ":   :              3"
   LOCATE 9, 38: PRINT ":   :             3"
   LOCATE 10, 18: PRINT "IMMMMMMMMMMMMMMMMMMMMJMMMJMMMMMMMMMMMMMMMOMMMM;"
   LOCATE 11, 12: PRINT "IMMMMM<"; : COLOR CI(2): PRINT "           AIR NAVIGATION/*RNAV*           ";
      COLOR CI(3): PRINT "HMMMMMM;"
   LOCATE 12, 12: PRINT ":IMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM;  Jonesboro Eagles  :"
   LOCATE 13, 12: PRINT "::"; : COLOR CI(1): PRINT "        by Robert A. Tims        ";
      COLOR CI(3): PRINT ":  "; : COLOR CI(4): PRINT "Civil Air Patrol"; : COLOR CI(3): PRINT "  :"
   LOCATE 14, 12: PRINT "::"; SPC(34); ":    "; : COLOR CI(4)
      PRINT "Squadron  HQ"; : COLOR CI(3): PRINT "    :"
   LOCATE 15, 12: PRINT "::              "; : COLOR 15: PRINT "DDBDD"; SPC(14);
```

```
      COLOR CI(3): PRINT ": ZDBD? IMMMM; ZDBD? :"
   LOCATE 16, 12: PRINT "::                    "; : COLOR 15: PRINT "3"; SPC(16);
      COLOR CI(3): PRINT ": CDED4 :     : CDED4 :"
   LOCATE 17, 12: PRINT "::                    "; : COLOR 15: PRINT "U08"; SPC(15);
      COLOR CI(3): PRINT ": @DADY :     : @DADY :"
   LOCATE 18, 12: PRINT "::       "; : COLOR 15: PRINT "@DDDDDmDRDA"; : COLOR 0: PRINT "i";
      COLOR 15: PRINT "ADRDmDDDDDY     "; : COLOR CI(3): PRINT ":        :   x:       :"
   LOCATE 19, 12: PRINT "::                  "; : COLOR 8: PRINT "k     k"; SPC(13);
      COLOR CI(3): PRINT ":        :     :       :"
   LOCATE 20, 1: COLOR CI(5), C(5): PRINT "000000"; : COLOR 8, 7: PRINT STRING$(68, 178);
      COLOR CI(5), C(5): PRINT "000000"
   LOCATE 21, 1: PRINT "0000"; : COLOR 8, 7: PRINT STRING$(72, 178); : COLOR CI(5), C(5): PRINT "0000"
   LOCATE 22, 1: PRINT "00"; : COLOR 8, 7: PRINT STRING$(76, 178); : COLOR CI(5), C(5): PRINT "00"
   LOCATE 23, 1: COLOR CI(0), 0
      PRINT SPC(14); "Copyright 1989 ABS Logic, Inc.  All rights reserved."; SPC(14);
   LOCATE 24, 1: PRINT SPC(21); "1616 Alonzo Ave., Jonesboro, AR  72401"; SPC(21);
   LOCATE 25, 1: PRINT SPC(19); "Parts copyrighted 1985-1988 Microsoft Corp."; SPC(18); : Time 10
END SUB

DEFINT A-Z
SUB LatFormat STATIC
   L = INSTR(Lat$, "."): Deg$ = LEFT$(Lat$, L - 1) + Degree$ + " " + MID$(Lat$, L + 1, 2)
   IF Format THEN
      Deg$ = Deg$ + "." + MID$(Lat$, L + 3, 1) + "'"
   ELSE
      Deg$ = Deg$ + "' " + MID$(Lat$, L + 3, 2) + CHR$(34)
   END IF
END SUB

DEFINT A-Z
SUB LatLngFormat STATIC
Start:
   Hue = 5: Heading$ = "LAT/LNG FORMAT": ScreenHeading: COLOR CI(Hue)
   LOCATE 5, 21: PRINT NRec; "Entries": COLOR CI(Hue): FormatDisplay
   LOCATE 8, 25: COLOR 0, C(Hue): PRINT " 0 "; : COLOR CI(Hue), 0: PRINT Text$
   LOCATE 11, 25: COLOR 0, C(Hue): PRINT " 1 "; : COLOR CI(Hue), 0: PRINT "  DEGREES, MINUTES, & SECONDS"
   LOCATE 13, 30: PRINT "Format:  DD"; DegreeSymbol$; " MM' SS"; CHR$(34);
   LOCATE 14, 35: COLOR C(Hue): PRINT "Example:  90"; DegreeSymbol$; " 38' 47"; CHR$(34)
   LOCATE 15, 30: COLOR CI(Hue): PRINT "Entry form:  DD.MMDD";
   LOCATE 16, 35: COLOR C(Hue): PRINT "Example:  90.3847"
   LOCATE 19, 25: COLOR 0, C(Hue): PRINT " 2 "; : COLOR CI(Hue), 0: PRINT "  DEGREES, MINUTES, & TENTHS"
   LOCATE 21, 30: PRINT "Format:  DD"; DegreeSymbol$; " MM.d"
   LOCATE 22, 35: COLOR C(Hue): PRINT "Example:  90"; DegreeSymbol$; " 38.8'"
   LOCATE 23, 30: COLOR CI(Hue): PRINT "Entry form:  DD.MMd";
   LOCATE 24, 35: COLOR C(Hue): PRINT "Example:  90.388";
   DO
      K$ = INPUT$(1)
      IF K$ = Esc$ OR K$ = Rtn$ OR K$ = "0" THEN EXIT SUB
   LOOP WHILE K$ < "1" OR K$ > "2": K = VAL(K$) - 1
   Wipe 8, 1: FOR n = 19 TO 24: Wipe n - K * 8, 2: NEXT
   COLOR C(Hue), 0: Row = 19 - K * 4: LOCATE Row, 21
   PRINT "Your choice is format option #"; K$; " Y/N? ";
   PRINT CHR$(222); : COLOR 16, C(Hue): PRINT "Y"; : COLOR C(Hue), 0: PRINT CHR$(221)
   DO
      K$ = UCASE$(INPUT$(1))
      IF K$ = "N" OR K$ = Esc$ THEN Wipe Row, 21: GOTO Start
   LOOP WHILE K$ <> "Y" AND K$ <> Rtn$
   IF K <> Format THEN
      Save = Format: Format = K
      IF NRec > 0 THEN
         CALL NavFile: Heading$ = "REFORMATTING LAT/LNG": ScreenHeading
         LOCATE 5, 21: PRINT NRec; "Entries": FormatDisplay: Escape "": TempNavFile
         FIELD 2, 2 AS Trec$, 3 AS Dum$, 8 AS TLat$, 9 AS TLng$: LSET Trec$ = MKI$(NRec): PUT 2, 1
         FOR Rec = 1 TO NRec
            IF INKEY$ = Esc$ THEN CLOSE : KILL "TempNav.DAT": Format = Save: EXIT SUB
            LOCATE 14, 29: PRINT "Reformatting Record"; Rec: GET 1, Rec + 1: A$ = Lat$: B$ = Lng$
            A = VAL(MID$(A$, 7, 2)): B = VAL(MID$(B$, 8, 2))
            IF K THEN
               T = A / 6: MID$(A$, 7, 2) = RIGHT$(STR$(T), 1) + " "
               IF T = 10 THEN
                  A = VAL(MID$(A$, 5, 2)): T = A + 1
                  IF T = 60 THEN S$ = " 0" ELSE S$ = STR$(T)
                  MID$(A$, 5, 2) = RIGHT$("0" + RIGHT$(S$, LEN(S$) - 1), 2)
                  IF T = 60 THEN
                     A = ABS(VAL(A$)): Sign = SGN(VAL(A$)): A = A + 1: S$ = STR$(A)
                     MID$(A$, 1, 3) = RIGHT$(" " + CHR$(45 + (Sign > -1) * 13) + RIGHT$(S$, LEN(S$) - 1), 3)
```

```
               END IF
            END IF
            T = B / 6: MID$(B$, 8, 2) = RIGHT$(STR$(T), 1) + " "
            IF T = 10 THEN
               B = VAL(MID$(B$, 6, 2)): T = B + 1
               IF T = 60 THEN S$ = " 0" ELSE S$ = STR$(T)
               MID$(B$, 6, 2) = RIGHT$("0" + RIGHT$(S$, LEN(S$) - 1), 2)
               IF T = 60 THEN
                  B = ABS(VAL(B$)): Sign = SGN(VAL(B$)): B = B + 1: S$ = STR$(B)
                  MID$(B$, 1, 4) = RIGHT$("  " + CHR$(45 + (Sign > -1) * 13) + RIGHT$(S$, LEN(S$) - 1), 4)
               END IF
            END IF
         ELSE
            S$ = STR$(A * 6): MID$(A$, 7, 2) = RIGHT$("0" + RIGHT$(S$, LEN(S$) - 1), 2)
            S$ = STR$(B * 6): MID$(B$, 8, 2) = RIGHT$("0" + RIGHT$(S$, LEN(S$) - 1), 2)
         END IF: LSET Dat$ = Old$: LSET TLat$ = A$: LSET TLng$ = B$: PUT 2, Rec + 1
      NEXT: OPEN "NavFile.OLD" FOR RANDOM AS 3: CLOSE : KILL "NavFile.OLD"
      NAME "NavFile.DAT" AS "NavFile.OLD": NAME "TempNav.DAT" AS "NavFile.DAT"
   END IF: SysFile: LSET Format$ = RIGHT$(STR$(Format), 1): PUT 3, 1: CLOSE
END IF
END SUB

DEFINT A-Z
SUB LngFormat STATIC
   L = INSTR(Lng$, "."): Deg$ = LEFT$(Lng$, L - 1) + Degree$ + " " + MID$(Lng$, L + 1, 2)
   IF Format THEN
      Deg$ = Deg$ + "." + MID$(Lng$, L + 3, 1) + "'"
   ELSE
      Deg$ = Deg$ + "' " + MID$(Lng$, L + 3, 2) + CHR$(34)
   END IF
END SUB

DEFINT A-Z
SUB Menu (UpperLimit) STATIC
   COLOR 7, 0: LOCATE 25, 32: PRINT "Which Function? "; CHR$(222);
   COLOR 16, 7: PRINT "0"; : COLOR 7, 0: PRINT CHR$(221); : K$ = ""
   WHILE K$ < "0" OR K$ > RIGHT$(STR$(UpperLimit), 1)
      K$ = INPUT$(1): IF K$ = Rtn$ THEN K$ = "0"
      IF K$ = Esc$ THEN
         CALL ScreenHeading: COLOR CI(0): K$ = "": LOCATE 15, 11
         PRINT "Quit the Air Navigation Program and return to DOS (Y/N)? ";
         COLOR C(0): PRINT CHR$(222); : COLOR 16, C(0): PRINT "N"; : COLOR C(0), 0: PRINT CHR$(221);
         WHILE K$ <> "N" AND K$ <> Rtn$ AND K$ <> Esc$
            K$ = INKEY$: IF K$ > "" THEN K$ = UCASE$(K$)
            IF K$ = "Y" THEN COLOR 7, 0: CLS : END
         WEND: K = 0: EXIT SUB
      END IF
   WEND: K = VAL(K$): LOCATE 25, 48: PRINT " "; K$; " ";
END SUB

DEFINT A-Z
SUB Navaid STATIC
   LOCATE 9, 43: COLOR CI(Hue), 0: PRINT Id$
   LOCATE 11, 26: LatFormat: COLOR 0, C(Hue)
   PRINT " 2 "; : COLOR CI(Hue), 0: PRINT "  Latitude: " + Deg$
   LOCATE 13, 26: LngFormat: COLOR 0, C(Hue)
   PRINT " 3 "; : COLOR CI(Hue), 0: PRINT "  Longitude: " + Deg$
   LOCATE 15, 26: VarFormat: COLOR 0, C(Hue)
   PRINT " 4 "; : COLOR CI(Hue), 0: PRINT "  Magnetic Variation: " + Deg$ + " "
   LOCATE 17, 26: COLOR 0, C(Hue)
   PRINT " 5 "; : COLOR CI(Hue), 0: PRINT "  Radio Frequency: " + Freq$
END SUB

DEFINT A-Z
SUB NavFile STATIC
   OPEN "NavFile.DAT" FOR RANDOM AS 1 LEN = 32
   FIELD 1, 1 AS AF$, 4 AS Id$, 8 AS Lat$, 9 AS Lng$, 3 AS Var$, 7 AS Freq$
   FIELD 1, 2 AS Rec$: GET 1, 1: NRec = CVI(Rec$): FileCheck
END SUB

DEFINT A-Z
SUB NavRec STATIC
   Rec = VAL(MID$(Index$, Ndx + 5, 3)): GET 1, Rec + 2: Index = INT(Ndx / 9)
END SUB

DEFINT A-Z
```

```
SUB Query (Row, Col, DataType, Length) STATIC
   Insert = 0
   IF INT(DataType / 10) THEN 'Check for Edit
      DataType = DataType - 10: LOCATE Row, Col
      IF DataType = 3 OR DataType = 5 THEN Flag = INSTR(Text$, "."): SignFlag = -(VAL(Text$) < 0)
      Text$ = Text$ + SPACE$(Length): Text$ = LEFT$(Text$, Length): Max = Length
      DO WHILE Max
         IF MID$(Text$, Max, 1) <> " " THEN EXIT DO ELSE Max = Max - 1
      LOOP
      IF Max THEN PRINT MID$(Text$, 1, Max);
      PRINT STRING$(Length - Max, "_"); : IF DataType = 4 THEN Flag = 1
      Max = Max + 1: Offset = 1: GOTO GetChar
   END IF
NewText:
   LOCATE Row, Col: Offset = 1: Flag = 0: Max = 1: SignFlag = 0
   PRINT STRING$(Length, 95); : Text$ = SPACE$(Length)
GetChar:
   IF Max = 1 THEN Msg$ = "abort" ELSE Msg$ = "erase"
   LOCATE 25, EscCol: COLOR CI(0), 0: PRINT Msg$; : COLOR CI(Hue), 0
   DO
      IF Length >= Offset AND (Flag = 0 OR DataType <> 3 OR Offset - Flag <> 5 - F) THEN
         S! = (TIMER - INT(TIMER)) * 2: S! = S! - INT(S!)
         IF S! >= .5 THEN LOCATE Row, Col + Offset - 1: PRINT CHR$(22 + 198 * Insert);
         IF S! < .5 THEN LOCATE Row, Col + Offset - 1: PRINT MID$(Text$, Offset, 1);
      END IF: K$ = INKEY$
   LOOP UNTIL K$ > CHR$(0)
   IF K$ = Rtn$ OR (K$ = Esc$ AND Max = 1) THEN 'Quit
      LOCATE 25, EscCol: COLOR CI(0), 0: PRINT "abort"; : COLOR CI(Hue), 0
      LOCATE Row, Col: PRINT Text$; : EXIT SUB
   END IF
   IF K$ = Esc$ THEN GOTO NewText
   IF K$ = CHR$(8) THEN 'Backspace
      IF Offset = 1 THEN GOTO GetChar ELSE Offset = Offset - 1: GOTO Delete
   END IF
   IF LEN(K$) = 2 THEN 'Check for extended key code
      Code = ASC(RIGHT$(K$, 1)) - 76: LOCATE Row, Col + Offset - 1: K$ = MID$(Text$, Offset, 1)
      IF Code = -5 THEN '[Home]
         IF Offset < Max THEN PRINT K$;  ELSE IF Offset < Length + 1 THEN PRINT "_";
         Offset = 1: GOTO GetChar
      END IF 'Then check for [End]
      IF Code = 3 AND Offset < Max THEN PRINT K$; : Offset = Max: GOTO GetChar
      IF Code = 6 AND DataType < 3 THEN Insert = Insert + (Insert = 1) - (Insert = 0): GOTO GetChar
      IF Code = 7 AND Offset < Max THEN
Delete:
         IF Offset = 1 THEN SignFlag = 0
         IF Flag = Offset THEN
            IF Flag <> Max - 1 THEN
               IF K$ = CHR$(8) THEN Offset = Offset + 1
               GOTO GetChar
            ELSE
               Flag = 0
            END IF
         END IF
         Temp$ = RIGHT$(Text$, Length - Offset)
         MID$(Text$, Offset, Length - Offset + 1) = Temp$ + " "
         LOCATE Row, Col + Offset - 1: Max = Max - 1
         IF Flag AND Offset < Flag THEN Flag = Flag - 1
         PRINT MID$(Temp$, 1, Max - Offset) + STRING$(Length + 1 - Max, "_"); : GOTO GetChar
      END IF 'Cursor [Left] or [Right]
      IF ABS(Code) <> 1 OR (Offset = Max AND Code = 1) THEN GOTO GetChar
      IF Offset < Max THEN PRINT K$;  ELSE IF Offset < Length + 1 THEN PRINT "_";
      IF (Offset > 1 OR Code = 1) AND (Offset < Length + 1 OR Code = -1) THEN Offset = Offset + Code
      GOTO GetChar
   END IF
   IF Offset > Length OR (K$ < " " OR K$ > CHR$(126)) OR K$ = CHR$(34) THEN GOTO GetChar
   ON DataType GOTO Caps, Number, LatLng, MagneticVariation, Decimal
KeepChar:
   LOCATE Row, Col + Offset - 1: PRINT K$;
   Temp$ = MID$(Text$, Offset, Length - Offset): MID$(Text$, Offset, 1) = K$
   IF Insert THEN
      Text$ = LEFT$(Text$, Offset) + Temp$: IF Max < Length + 1 THEN Max = Max + 1
      IF Max > 2 THEN LOCATE Row, Col: PRINT LEFT$(Text$, Max - 1)
   END IF
   Offset = Offset + 1: IF Offset - Max = 1 THEN Max = Offset
   GOTO GetChar
```

```
Caps: 'Data Type 1
   K$ = UCASE$(K$): GOTO KeepChar

Number: 'Data Type 2
   IF K$ < "0" OR K$ > "9" THEN GOTO GetChar ELSE GOTO KeepChar

LatLng: 'Data Type 3 - Insert disabled
   IF Offset = 1 THEN
      IF K$ = "-" OR K$ = "+" THEN SignFlag = 1: GOTO KeepChar
      IF SignFlag THEN GOTO GetChar
   END IF
   IF Flag = 0 AND K$ = "." AND Max - Offset < 6 - Format THEN Flag = Offset: GOTO KeepChar
   IF (Flag AND Offset - Flag > 4 - F) OR (Flag = 0 AND Offset > Length - 6 + Format + SignFlag) THEN
      GOTO GetChar
   END IF: IF K$ < "0" OR K$ > "9" THEN GOTO GetChar
   IF Offset = Flag THEN
      IF (Max = Offset + 1 AND Offset < Length - 5 + Format + SignFlag) THEN
         Flag = 0
      ELSE
         GOTO GetChar
      END IF
   END IF: GOTO KeepChar

MagneticVariation: 'Data Type 4 - Insert disabled & Middle-of-string backspace disabled before direction
   K$ = UCASE$(K$)
   IF (Flag = 0 OR Offset = Flag) THEN
      IF (K$ = "-" OR K$ = "+" OR K$ = "E" OR K$ = "W") THEN Flag = Offset: GOTO KeepChar
   END IF
   IF K$ > "/" AND K$ < ":" AND Flag <> Offset THEN
      IF (Flag = 1 OR Offset < Flag OR (Offset < 3 AND Flag = 0)) THEN GOTO KeepChar
   END IF: GOTO GetChar

Decimal: 'Data Type 5 - One Decimal place allowed - Insert disabled and backspace modified
   IF Flag = 0 AND K$ = "." AND Max - Offset < 3 THEN Flag = Offset: GOTO KeepChar
   IF (Flag AND Offset - Flag > 1) OR (Flag = 0 AND Offset > Length - 2) THEN GOTO GetChar
   IF K$ < "0" OR K$ > "9" THEN GOTO GetChar
   IF Offset = Flag THEN
      IF Offset < Length - 1 AND (Max = Offset + 1 OR Max < Length AND Offset < Length - 1) THEN
         Flag = 0
      ELSE
         GOTO GetChar
      END IF
   END IF: GOTO KeepChar
END SUB

DEFINT A-Z
SUB RouteScreen (WaypointFlag) STATIC
   LOCATE 9, 32: COLOR 0, C(Hue): PRINT " 1 "; : COLOR CI(Hue), 0: PRINT "  NAVAID"
   LOCATE 11, 32: COLOR 0, C(Hue): PRINT " 2 "; : COLOR CI(Hue), 0: PRINT "  AIRPORT"
   LOCATE 13, 32: COLOR 0, C(Hue): PRINT " 3 "; : COLOR CI(Hue), 0: PRINT "  INTERSECTION"
   IF WaypointFlag THEN
      LOCATE 15, 32: COLOR 0, C(Hue): PRINT " 4 "; : COLOR CI(Hue), 0: PRINT "  WAYPOINT"
      LOCATE 19, 36: PRINT "Next Fix? ";
   ELSE
      LOCATE 19, 32: PRINT "Departure Point? ";
   END IF: COLOR CI(Hue) + 16: PRINT CHR$(22): Escape "entry"
   DO
      K$ = INPUT$(1): IF K$ = Esc$ OR K$ = Rtn$ THEN EXIT SUB
   LOOP WHILE VAL(K$) < 1 OR VAL(K$) > 3 + WaypointFlag
   K = VAL(K$): Id = -(K - 1) * (K < 3) - 3 * (K = 3) - 2 * (K = 4)
END SUB

DEFINT A-Z
SUB ScreenHeading STATIC
      COLOR 0, 0, 0: CLS : LOCATE 1, 30: COLOR 15: PRINT "AIR NAVIGATION/*RNAV*"
      LOCATE 2, CINT((82 - LEN(Heading$)) / 2): PRINT Heading$
      LOCATE 3, 21: PRINT STRING$(40, 196): COLOR CI(Hue)
END SUB

DEFINT A-Z
SUB Strip STATIC
   Length = 0
   DO WHILE Length < LEN(Text$)
      IF MID$(Text$, Length + 1, 1) <> " " THEN Text$ = RIGHT$(Text$, LEN(Text$) - Length): EXIT DO
      Length = Length + 1
   LOOP
```

```
   DO WHILE LEN(Text$)
      IF MID$(Text$, LEN(Text$), 1) <> " " THEN EXIT DO
      Text$ = MID$(Text$, 1, LEN(Text$) - 1)
   LOOP
END SUB

SUB SysFile STATIC
   SHARED CFlg$, Format$, Pr1$, Pr2$, SL$, DL$
   OPEN "NAVSYS.DAT" FOR RANDOM AS 3
   FIELD 3, 1 AS CFlg$, 1 AS Format$, 24 AS Pr1$, 24 AS Pr2$, 1 AS SL$, 1 AS DL$
   IF LOF(3) = 0 THEN FIELD 3, 128 AS A$: LSET A$ = "00" + STRING$(48, 0) + "-=": PUT 3, 1
   GET 3, 1
END SUB

DEFINT A-Z
SUB TempNavFile
   OPEN "TempNav.DAT" FOR RANDOM AS 2: CLOSE 2: KILL "TempNav.DAT"
   OPEN "TempNav.DAT" FOR RANDOM AS 2 LEN = 32: FIELD 2, 32 AS Dat$: FIELD 1, 32 AS Old$
END SUB

SUB Time (Seconds) STATIC
   T! = TIMER: WHILE TIMER - (T! - T! * LEN(INKEY$)) < Seconds: WEND
END SUB

SUB VarFormat STATIC
   Deg$ = RIGHT$(STR$(100 + ABS(VAL(Var$))), 2) + Degree$ + " "
   IF ASC(Deg$) = 48 THEN MID$(Deg$, 1, 1) = " "
   IF VAL(Var$) < 0 THEN Deg$ = Deg$ + "E" ELSE Deg$ = Deg$ + "W"
END SUB

DEFINT A-Z
SUB Wipe (Row, Col) STATIC
   COLOR , 0: LOCATE Row, Col: PRINT SPC(80 - Col + (Row > 23));
END SUB

FUNCTION Zero$ (A$) STATIC
   Zero$ = RIGHT$("00" + RIGHT$(STR$(ASC(A$)), LEN(STR$(ASC(A$))) - 1), 3)
END FUNCTION
```

```
'AirNavigation Nav Data
'File Conversion from old format to new format
'Copyright 1989 Robert A. Tims & ABS Logic, Inc.
'1-20-89 Version Date - Program File Name: NFILEFIX.BAS

   DEFINT A-Z: CLS : CONST No = 0, Yes = NOT No
   LOCATE 2, 12: PRINT "Stand by - Converting your old Nav.Dat file to NavFile.Dat"

'Old Nav Data File
   OPEN "NAV.DAT" FOR RANDOM AS 1 LEN = 23
   FIELD 1, 1 AS OAF$, 4 AS OId$, 7 AS OLat$, 8 AS OLng$, 3 AS OVar$
   FIELD 1, 2 AS Rec$: GET 1, 1: NRec = CVI(Rec$)
   OPEN "NAVDATA.NDX" FOR INPUT AS 2: INPUT #2, Ndx$: CLOSE 2
   IF NRec <> LEN(Ndx$) / 8 THEN
      LOCATE 14, 14: PRINT "The Nav Data needs to be re-indexed before continuing!": END
   END IF: LOCATE 5, 26: PRINT "Total records to be moved:"; NRec

'New Nav Data File
   OPEN "NavFile.DAT" FOR RANDOM AS 2 LEN = 32
   FIELD 2, 1 AS AF$, 4 AS Id$, 8 AS Lat$, 9 AS Lng$, 3 AS Var$, 7 AS Freq$
   FIELD 2, 2 AS NoRec$: Index$ = "": Rec = -1: LOCATE 12, 36: PRINT "Moving "
   LOCATE 14, 30: PRINT "from Nav.Dat record": LOCATE 16, 29: PRINT "to NavFile.Dat record"

   FOR I = 1 TO NRec
      Rec$ = MID$(Ndx$, (I * 8) - 3, 3): GET 1, VAL(Rec$) + 2
      LSET AF$ = OAF$: LSET Id$ = OId$: LSET Lat$ = " " + OLat$
      LOCATE 12, 43: PRINT OId$: LOCATE 14, 50: PRINT VAL(Rec$); "    ": LOCATE 16, 51: PRINT Rec + 2; "    "
      LSET Lng$ = " " + OLng$: LSET Var$ = OVar$: LSET Freq$ = SPACE$(7)
      Rec = Rec + 1: PUT 2, Rec + 2
      Index$ = Index$ + AF$ + Id$ + RIGHT$(STR$(1000 + Rec), 3) + CHR$(254)
   NEXT
   IF NOT Rec THEN
      GET 2, 1: LSET NoRec$ = MKI$(Rec + 1): PUT 2, 1: CLOSE
      OPEN "NavFile.NDX" FOR OUTPUT AS 1: PRINT #1, Index$
   END IF: CLOSE : END
```

```
'AirNavigation Nav Data
'File Conversion from old format to new format
'Copyright 1989 Robert A. Tims & ABS Logic, Inc.
'1-20-89 Version Date - Program File Name: NFILEFIX.BAS

   DEFINT A-Z: CLS : CONST No = 0, Yes = NOT No
   LOCATE 2, 12: PRINT "Stand by - Converting your old Nav.Dat file to NavFile.Dat"

'Old Nav Data File
   OPEN "NAV.DAT" FOR RANDOM AS 1 LEN = 23
   FIELD 1, 1 AS OAF$, 4 AS OId$, 7 AS OLat$, 8 AS OLng$, 3 AS OVar$
   FIELD 1, 2 AS Rec$: GET 1, 1: NRec = CVI(Rec$)
   OPEN "NAVDATA.NDX" FOR INPUT AS 2: INPUT #2, Ndx$: CLOSE 2
   IF NRec <> LEN(Ndx$) / 8 THEN
      LOCATE 14, 14: PRINT "The Nav Data needs to be re-indexed before continuing!": END
   END IF: LOCATE 5, 26: PRINT "Total records to be moved:"; NRec

'New Nav Data File
   OPEN "NavFile.DAT" FOR RANDOM AS 2 LEN = 32
   FIELD 2, 1 AS AF$, 4 AS Id$, 8 AS Lat$, 9 AS Lng$, 3 AS Var$, 7 AS Freq$
   FIELD 2, 2 AS NoRec$: Index$ = "": Rec = -1: LOCATE 12, 36: PRINT "Moving "
   LOCATE 14, 30: PRINT "from Nav.Dat record": LOCATE 16, 29: PRINT "to NavFile.Dat record"

   FOR I = 1 TO NRec
      Rec$ = MID$(Ndx$, (I * 8) - 3, 3): GET 1, VAL(Rec$) + 2
      LSET AF$ = OAF$: LSET Id$ = OId$: LSET Lat$ = " " + OLat$
      LOCATE 12, 43: PRINT OId$: LOCATE 14, 50: PRINT VAL(Rec$); "    ": LOCATE 16, 51: PRINT Rec + 2; "    "
      LSET Lng$ = " " + OLng$: LSET Var$ = OVar$: LSET Freq$ = SPACE$(7)
      Rec = Rec + 1: PUT 2, Rec + 2
      Index$ = Index$ + AF$ + Id$ + RIGHT$(STR$(1000 + Rec), 3) + CHR$(254)
   NEXT
   IF NOT Rec THEN
      GET 2, 1: LSET NoRec$ = MKI$(Rec + 1): PUT 2, 1: CLOSE
      OPEN "NavFile.NDX" FOR OUTPUT AS 1: PRINT #1, Index$
   END IF: CLOSE : END
```

```
DEFINT A-Z
SUB Facility STATIC
   Fac$ = "NAVAID ": IF Id THEN Fac$ = "AIRPORT": IF Id < 1 THEN Fac$ = "DELETED"
   IF Id > 1 THEN Fac$ = "WAYPOINT": IF Id = 3 THEN Fac$ = "INTERSECTION"
'NOTES:
   '0 = NAVAID
   '1 = AIRPORT
   '2 = WAYPOINT
   '3 = INTERSECTION
   '4 = TERMINAL WAYPOINT
   '5 = END OF FLIGHT
END SUB

DEFINT A-Z
FUNCTION Increment (Index, CoordinateFlag) STATIC
   Increment = Index + 6 + CoordinateFlag * 27 + 15 * -(Id = 3)
'NOTES: Flight$ (Flight route data)
   'Navaid, Airport, or Waypoint -
      'Byte 1:                  Fix identifier (see Sub Facility)
      'Bytes 2 to 5:            Four letter identifier
      'Byte 6:                  Coordinate Flag (0 = in NavFile, 1 = in Flight$)
      'Bytes 7 to 33:           Coordinate data if stored in the Flight$
      '  Bytes 7 to 14:            Latitude (8 bytes - bytes 1 to 8 relative)
      '  Bytes 15 to 23:           Longtitude (9 bytes - bytes 9 to 17 relative)
      '  Bytes 24 to 26:           Variation (3 bytes - bytes 18 to 20 relative)
      '  Bytes 27 to 33:           Frequence (7 bytes - bytes 21 to 27 relative)
   'Intersection -
      'Byte 1:                  "3" for intersection
      'Bytes 2 to 6:            Five letter identifier
      'Bytes 7 to 11(38):       Reference Navaid Data
      '  Bytes 7 to 10:            Four letter identifier
      '  Byte 11:                  Coordinate Flag
      '  Bytes (12) to (38):       Coordinate data if stored in the Flight$
      'Bytes 12(39) to 16(43):  Radial (5 bytes - 000.1 to 360.0)
      'Bytes 17(44) to 21(48):  DME Distance (5 bytes - 0.0 to 999.9)
END FUNCTION

DEFINT A-Z
SUB FlightDisplay STATIC
   Heading$ = "FLIGHT ROUTE": Hue = 2: ScreenHeading: Escape "listing": Row = 7: Index = 1: Departure = Yes
ListNav:
   Id = VAL(MID$(Flight$, Index, 1)): Facility
   IF Id < 4 THEN
      K$ = INKEY$: IF K$ = Esc$ THEN EXIT SUB
      LOCATE Row, 26: Row = Row + 1
      IF Row = 23 THEN
         Msg$ = "Partial listing from " + Departure$: LOCATE 5, 28: PRINT Msg$
         Continue (23): IF K$ = Esc$ THEN EXIT SUB
         CALL ScreenHeading: LOCATE 5, 28: PRINT Msg$: Escape "listing"
         LOCATE 7, 26: PRINT LastFix$: Row = 9: LOCATE 8, 26
      END IF
   END IF
   ON Id GOTO Id1, Id1, Id3, Id4, Id5
Id1:
   Text$ = MID$(Flight$, Index + 1, 4): Strip: LastFix$ = " " + RIGHT$("    " + Text$, 4) + " " + Fac$
   PRINT LastFix$; : Index = Increment(Index, VAL(MID$(Flight$, Index + 5, 1))): GOTO DepartCk
Id3:
   Text$ = MID$(Flight$, Index + 1, 5): Strip: PRINT RIGHT$("     " + Text$, 5) + " " + Fac$
   Index = Increment(Index, VAL(MID$(Flight$, Index + 10, 1)))
DepartCk:
   IF Departure THEN Departure$ = Text$: Departure = No
   GOTO ListNav
Id4:
   PRINT " * " + Fac$ + " " + MID$(Flight$, Index + 2, 3)
   Index = Increment(Index, VAL(MID$(Flight$, Index + 5, 1))): GOTO ListNav
Id5:
   Destination$ = Text$: Heading$ = Departure$ + " to " + Destination$
   Wipe 5, 28: LOCATE 5, 41 - LEN(Heading$) / 2: PRINT Heading$: Direction = 1
   LOCATE 23, 28: COLOR 7, 0: PRINT "Is this correct (Y/N)? "; CHR$(222);
   COLOR 16, 7: PRINT "Y"; : COLOR 7, 0: PRINT CHR$(221); : Escape "routing"
   DO
      K$ = UCASE$(INPUT$(1))
   LOOP WHILE K$ <> Esc$ AND K$ <> "N" AND K$ <> "Y" AND K$ <> Rtn$
END SUB
```

```
'AirNavigation Nav Data
'File Conversion from old format to new format
'Copyright 1989 Robert A. Tims & ABS Logic, Inc.
'1-20-89 Version Date - Program File Name: NFILEFIX.BAS

   DEFINT A-Z: CLS : CONST No = 0, Yes = NOT No
   LOCATE 2, 12: PRINT "Stand by - Converting your old Nav.Dat file to NavFile.Dat"

'Old Nav Data File
   OPEN "NAV.DAT" FOR RANDOM AS 1 LEN = 23
   FIELD 1, 1 AS OAF$, 4 AS OId$, 7 AS OLat$, 8 AS OLng$, 3 AS OVar$
   FIELD 1, 2 AS Rec$: GET 1, 1: NRec = CVI(Rec$)
   OPEN "NAVDATA.NDX" FOR INPUT AS 2: INPUT #2, Ndx$: CLOSE 2
   IF NRec <> LEN(Ndx$) / 8 THEN
      LOCATE 14, 14: PRINT "The Nav Data needs to be re-indexed before continuing!": END
   END IF: LOCATE 5, 26: PRINT "Total records to be moved:"; NRec

'New Nav Data File
   OPEN "NavFile.DAT" FOR RANDOM AS 2 LEN = 32
   FIELD 2, 1 AS AF$, 4 AS Id$, 8 AS Lat$, 9 AS Lng$, 3 AS Var$, 7 AS Freq$
   FIELD 2, 2 AS NoRec$: Index$ = "": Rec = -1: LOCATE 12, 36: PRINT "Moving "
   LOCATE 14, 30: PRINT "from Nav.Dat record": LOCATE 16, 29: PRINT "to NavFile.Dat record"

   FOR I = 1 TO NRec
      Rec$ = MID$(Ndx$, (I * 8) - 3, 3): GET 1, VAL(Rec$) + 2
      LSET AF$ = OAF$: LSET Id$ = OId$: LSET Lat$ = " " + OLat$
      LOCATE 12, 43: PRINT OId$: LOCATE 14, 50: PRINT VAL(Rec$); "    ": LOCATE 16, 51: PRINT Rec + 2; "    "
      LSET Lng$ = " " + OLng$: LSET Var$ = OVar$: LSET Freq$ = SPACE$(7)
      Rec = Rec + 1: PUT 2, Rec + 2
      Index$ = Index$ + AF$ + Id$ + RIGHT$(STR$(1000 + Rec), 3) + CHR$(254)
   NEXT
   IF NOT Rec THEN
      GET 2, 1: LSET NoRec$ = MKI$(Rec + 1): PUT 2, 1: CLOSE
      OPEN "NavFile.NDX" FOR OUTPUT AS 1: PRINT #1, Index$
   END IF: CLOSE : END
```

```
100 'AIR NAVIGATION/*RNAV* - Basic Source Code - Copyright 1986 ROBERT A. TIMS & ABS Logic, Inc. 10-10-86
130 ON ERROR GOTO 400:PRINT CHR$(27)"[0;30;40m"CHR$(27)"[2J":DEFINT A-Z:WIDTH "LPT1:",255:WIDTH 255
140 GOSUB 20200:CFLG=VAL(CFLG$):PRINIT$=PR1$:PRDEG$=PR2$:F=VAL(F$):PRINT CHR$(27)"[="CFLG$"h"CHR$(27)"[2J"
150 CLOSE 3:DIM C$(5),B$(5):C$="777777":IF CFLG THEN C$="143526"
160 FOR I=0 TO 5:A$=MID$(C$,I+1,1):C$(I)="3"+A$:B$(I)=";4"+A$+"m":NEXT:IF INKEY$<>CHR$(27) THEN GOSUB 30000
170 ON ERROR GOTO 20300:OPEN "NAVDATA.NDX" FOR INPUT AS 2:INPUT #2,INDEX$:ON ERROR GOTO 400:CLOSE 2
180 GOSUB 20000:A=LOF(1):CLOSE 1
200 PI#=3.1415926535897938#:C#=PI#/2:CR#=2*PI#:DR#=PI#/180:RD#=180/PI#:RS#=RD#*60:D$="x"
210 CSR$=" "+CHR$(27)+"[1D"
220 MENU$=CHR$(27)+"[0;37m"+CHR$(27)+"[24;12HWhich Function? "+CSR$
240 DEF FNASIN#(X#)=ATN(X#/SQR(1-X#*X#))
250 DEF FNACOS#(X#)=C#-FNASIN#(X#)
260 DEF FNCAP$(A$)=CHR$(ASC(A$)+32*(A$>="a" AND A$<="z"))
270 DEF FNID$(A$)=LEFT$(RIGHT$(A$,LEN(A$)+(ASC(A$)=32))+" ",LEN(A$))
280 DEF FNZERO$(A$)=" "+RIGHT$("00"+RIGHT$(STR$(ASC(A$)),LEN(STR$(ASC(A$)))-1),3)
290 IF A=0 THEN K=0:GOSUB 9000
300 C$="37":H$="MAIN MENU":GOSUB 10000
310 X$="9":Y$="12":C$=C$(1)+";1":B$=B$(1):A$="@ 1 A  ROUTE ENTRY":GOSUB 10200
320 X$="11":C$=C$(2)+";1":B$=B$(2):A$="@ 2 A  LOAD FLIGHT":GOSUB 10200
330 X$="13":C$=C$(3)+";1":B$=B$(3):A$="@ 3 A  AIRNAV DATA":GOSUB 10200
340 X$="16":C$=C$(4):B$=B$(4):A$="@ 4 A  Printer Codes":GOSUB 10200
350 X$="18":C$=C$(5):B$=B$(5):A$="@ 5 A  Screen Setup":GOSUB 10200
360 X$="21":C$=C$(0)+";1":B$=B$(0):A$="@ 0 A  QUIT":GOSUB 10200:PRINT MENU$;
370 K$=INPUT$(1):IF K$<"0" OR K$>"5" THEN 370 ELSE K=VAL(K$):ON K GOTO 2000,5000,6000,8000,8500
400 PRINT CHR$(27)"[="RIGHT$(STR$(CFLG+2),1)"h"CHR$(27)"[0m"CHR$(27)"[2J":END
500 IF FLG2 THEN A$="":GOSUB 590 ELSE GOSUB 10400 'DATA ENTRY ***
510 A=1:PRINT CHR$(27)"[u"CHR$(27)"["C$"m"STRING$(L,95);:A$=SPACE$(L):D=0
520 Y$=";"+RIGHT$(STR$(Y+A+100),2)+"H":IF L>=A THEN PRINT CHR$(27)"[0;"C$"m"CHR$(27)"["X$Y$CSR$;
530 K$=FNCAP$(INPUT$(1)):IF A=1 AND (K$=CHR$(13) OR K$=CHR$(27)) THEN 580 ELSE IF K$=CHR$(13) THEN 590 ELSE IF K$=
CHR$(27) THEN 500
540 IF K$=CHR$(8) THEN IF A=1 THEN 520 ELSE A=A-1:MID$(A$,A,1)=" ":IF A=L THEN 520 ELSE PRINT CHR$(27)"["X$Y$"_"::
GOTO 520
550 IF A>L OR (K$<" " OR K$>CHR$(126)) THEN 520
560 ON N GOTO 600,620,700,620
570 PRINT CHR$(27)"["X$Y$K$;:MID$(A$,A,1)=K$:A=A+1:GOTO 520
580 A$=S$
590 PRINT CHR$(27)"["X$";"RIGHT$(STR$(Y+101),2)"H"CHR$(27)"[s"A$:RETURN
600 'NUMBERS ***
610 IF K$<"0" OR K$>"9" THEN 520 ELSE 570
620 'DECIMAL ***
630 IF A<=D THEN D=0
640 IF D=0 AND K$="." THEN D=A:GOTO 570
650 IF N=2 THEN IF (D AND A-D>4-F) OR (D=0 AND A>L-5+F) THEN 520 ELSE 600
660 IF (D AND A-D>1) OR (D=0 AND A>L-2) THEN 520 ELSE 600
700 'VAR ***
710 IF A<=0 THEN D=0
720 IF K$>"/" AND K$<":" AND (D=1 OR (A<3 AND D=0)) THEN 570
730 IF D OR NOT(K$="-" OR K$="+" OR K$="E" OR K$="W") THEN 520 ELSE D=A:GOTO 570
900 'FACILITY ID ***
910 FAC$="NAVAID":IF ID THEN FAC$="AIRPORT":IF ID>1 THEN FAC$="WAYPOINT":IF ID=3 THEN FAC$="INTERSECTION"
920 RETURN
1400 'ID INPUT ***
1410 X$="12":Y$="7":Y=1:GOSUB 10400:A$="@ 1 A Identifier:":GOSUB 10200
1420 N=0:Y=22:L=3+ID:IF ID=3 THEN L=5
1430 GOSUB 500:GOSUB 10300:IF L THEN WHILE ASC(A$)=32:A$=RIGHT$(A$,LEN(A$)-1):WEND
1440 IF A=1 THEN RETURN ELSE IF L=0 THEN IF E THEN GOSUB 11000:GOTO 1420 ELSE A=1:RETURN
1450 L=LEN(A$):ID1$=A$:IF ID=3 THEN RETURN ELSE IF ID=0 OR L<4 THEN A$=" "+LEFT$(A$+"   ",3)
1460 I$=RIGHT$(A$,3):INDEX=INSTR(INDEX$,AF$+I$):IF INDEX=0 THEN LSET ID$=A$:IF E1 THEN FLG1=1:MID$(INDEX$,RECNDX,3
)=I$
1470 IF INDEX AND E THEN GOSUB 11050:GOTO 1420 ELSE A$=FNID$(A$):GOTO 590
1500 'LAT INPUT ***
1510 X$="14":Y$="7":Y=1:GOSUB 10400:A$="@ 2 A Latitude:":GOSUB 10200:Y=20:L=7-F
1520 N=2:GOSUB 500:IF A=1 AND E THEN GOSUB 1550:GOTO 590 ELSE IF A=1 THEN RETURN
1530 L#=VAL(A$)*10000+1000000!:A$=STR$(L#):A$=MID$(A$,3,2)+"."+MID$(A$,5,4-F):IF ASC(A$)=48 THEN MID$(A$,1,1)=" "
1540 IF L#>1900000! OR VAL(MID$(A$,4,2))>59 OR (F=0 AND VAL(MID$(A$,6,2))>59) THEN GOSUB 11000:GOTO 1520
1550 LSET LAT$=A$:GOSUB 1570:IF E THEN FLG1=1:GOTO 590 ELSE 590
1570 L=INSTR(LAT$,"."):DEG$=LEFT$(LAT$,L-1)+D$+" "+MID$(LAT$,L+1,2)
1580 IF F THEN DEG$=DEG$+"."+MID$(LAT$,L+3,1)+"'" ELSE DEG$=DEG$+"' "+MID$(LAT$,L+3,2)+CHR$(34)
1590 A$=DEG$:RETURN
1600 'LNG INPUT ***
1610 X$="16":Y$="7":Y=1:GOSUB 10400:A$="@ 3 A Longitude:":GOSUB 10200:Y=21:L=8-F
1620 N=2:GOSUB 500:IF A=1 AND E THEN GOSUB 1650:GOTO 590 ELSE IF A=1 THEN RETURN
1630 L#=VAL(A$)*10000+10000000#:A$=STR$(L#):A$=MID$(A$,3,3)+"."+MID$(A$,6,4-F):IF MID$(A$,1,1)="0" THEN MID$(A$,1,
1)=" ":IF MID$(A$,2,1)="0" THEN MID$(A$,2,1)=" "
1640 IF L#>11800000# OR VAL(MID$(A$,5,2))>59 OR (F=0 AND VAL(MID$(A$,7,2))>59) THEN GOSUB 11000:GOTO 1620
1650 LSET LNG$=A$:GOSUB 1670:IF E THEN FLG1=1:GOTO 590 ELSE 590
```

```
1670 L=INSTR(LNG$,"."):DEG$=LEFT$(LNG$,L-1)+D$+" "+MID$(LNG$,L+1,2)
1680 IF F THEN DEG$=DEG$+"."+MID$(LNG$,L+3,1)+"'" ELSE DEG$=DEG$+"' "+MID$(LNG$,L+3,2)+CHR$(34)
1690 A$=DEG$:RETURN
1700 'VAR INPUT ***
1710 X$="18":Y$="7":Y=1:GOSUB 10400:A$="@ 4 A Magnetic Variation:":GOSUB 10200:Y=30:L=3:N=3:GOSUB 500:IF A=1 AND E
 THEN GOSUB 1750:GOTO 590 ELSE IF A=1 THEN RETURN
1720 IF ASC(A$)>47 AND ASC(A$)<59 THEN VAR=VAL(A$) ELSE VAR=VAL(RIGHT$(A$,2))
1730 IF INSTR(A$,"E") OR INSTR(A$,"-") THEN VAR=-VAR
1740 LSET VAR$=STR$(VAR):GOSUB 1750:IF E THEN FLG1=1:GOTO 590 ELSE 590
1750 DEG$=RIGHT$(STR$(VAR),LEN(STR$(VAR))-1)+D$+" ":IF VAR<0 THEN DEG$=DEG$+"E" ELSE DEG$=DEG$+"W"
1760 A$=DEG$:RETURN
1800 'ROUTE SCREEN ***
1810 X$="12":Y$="12":C$=CS$:A$="@ 1 A  NAVAID":GOSUB 10200
1820 X$="14":A$="@ 2 A  AIRPORT":GOSUB 10200
1830 X$="16":A$="@ 3 A  INTERSECTION":GOSUB 10200
1840 IF FLG3 THEN GOTO 10500 ELSE X$="18":A$="@ 4 A  WAYPOINT":GOSUB 10200:GOTO 10500
1900 Y=0:X$="6":GOSUB 10400:X$="8":GOSUB 10400:GOSUB 6350:A$=M$+FAC$:X$="8":C$=CS$:GOTO 10100
2000 'ROUTE ENTRY ***
2010 GOSUB 20000:H$="ROUTE PLANNING"
2020 FLG3=1:FLG5=1:C$=C$(1):CS$=C$+";1":B$=B$(1):GOSUB 10000:GOSUB 1800
2030 FLG=1:FLG1=0:FLG2=0:PRINT CHR$(27)"[0;"C$"m"CHR$(27)"[22;12HDeparture Point? "CSR$;
2040 K$=INPUT$(1):IF K$=CHR$(27) OR K$=CHR$(13) THEN CLOSE 1:GOTO 300 ELSE IF K$<"1" OR K$>"3" THEN 2040
2050 K=VAL(K$):ID=K-1:IF K=3 THEN ID=3
2100 GOSUB 900:M$="DEPARTURE ":GOSUB 1900
2120 FX$=RIGHT$(STR$(ID),1):LSET AF$=FX$:E=0:E1=0:S$="":GOSUB 1400
2130 IF A=1 THEN 2020
2140 IF K<3 THEN GOSUB 3000:ID2$=FNID$(ID1$) ELSE GOSUB 3100
2150 IF A=1 THEN 2130 ELSE FLT$=FX$:FLG4=0
2160 GOSUB 3300
2200 FLG3=0:FAC$="":M$="LAST FIX WAS "+ID2$:GOSUB 1900:GOSUB 1800
2210 PRINT CHR$(27)"[0;"C$"m"CHR$(27)"[22;15HNext Fix? ";CSR$;
2220 K$=INPUT$(1):IF K$=CHR$(13) AND FLG4 THEN 2350 ELSE IF K$=CHR$(13) OR K$=CHR$(27) THEN GOSUB 6400:GOTO 300
2230 IF K$<"1" OR K$>"4" THEN 2220 ELSE K=VAL(K$):ID=K-1:IF K=3 THEN ID=3 ELSE IF K=4 THEN ID=2
2300 GOSUB 900:M$="ENROUTE ":GOSUB 1900:FX$=RIGHT$(STR$(ID),1):LSET AF$=FX$:IF K=4 THEN ID=0:LSET AF$="0":GOSUB 90
0
2310 GOSUB 1400:IF A=1 THEN 2340
2320 IDS$=ID2$:IF ID<3 THEN GOSUB 3000:ID2$=FNID$(ID1$) ELSE GOSUB 3100
2330 IF A>1 THEN FLT$=FLT$+FX$:FLG4=1 ELSE ID2$=IDS$
2340 GOTO 2200
2350 GOSUB 3300
2360 FLT$=FLT$+"5":GOSUB 6400:GOSUB 3400:CLOSE 1:IF K$=CHR$(27) THEN 300 ELSE IF K$="N" THEN 2000
2400 'ROUTE MENU ***
2410 C$=C$(1):B$=B$(1):H$=IDF$+" to "+IDT$:GOSUB 10000:P=0
2420 X$="10":Y$="15":C$=C$(5)+";1":B$=B$(5):A$="@ 1 A  VIEW":GOSUB 10200
2430 X$="12":C$=C$(4)+";1":B$=B$(4):A$="@ 2 A  PRINT":GOSUB 10200
2440 X$="14":C$=C$(3)+";1":B$=B$(3):A$="@ 3 A  REVERSE":GOSUB 10200
2450 X$="16":C$=C$(2)+";1":B$=B$(2):A$="@ 4 A  SAVE":GOSUB 10200
2460 X$="19":C$=C$(0)+";1":B$=B$(0):A$="@ 0 A  MENU":GOSUB 10200:PRINT MENU$;
2470 K$=INPUT$(1):IF K$="0" THEN 300 ELSE IF K$<"1" OR K$>"4" THEN 2470 ELSE K=VAL(K$): C$=C$(5):ON K GOTO 4000,25
00,2550,5500
2500 'SET UP FOR PRINT ***
2510 P=1:C$=C$(4):GOSUB 10000:X$="13":Y$="2":A$="Prepare Printer {top-of-formzzon-line}":GOSUB 10200
2520 GOSUB 10540:X$="15":GOSUB 11200:IF K$=CHR$(27) THEN 2400 ELSE ON ERROR GOTO 7520:GOTO 4000
2550 SWAP IDF$,IDT$:H$=IDF$+" to "+IDT$:RD=-RD:GOSUB 10080:GOTO 2470
2600 'GET FILE DATA ***
2610 FLG=VAL(MID$(FLT$,A+5,1)):I$=MID$(FLT$,A+1,4)
2620 IF FLG THEN LSET LAT$=MID$(FLT$,A+6,7):LSET LNG$=MID$(FLT$,A+13,8):LSET VAR$=MID$(FLT$,A+21,3):GOTO 2640
2630 INDEX=INSTR(INDEX$,I$):IF INDEX=0 THEN 20100 ELSE REC=VAL(MID$(INDEX$,INDEX+4,3)):GET 1,REC+2:I$=ID$
2640 VAR2#=VAL(VAR$)*DR#
2700 'CONVERT LAT/LNG TO RADIANS
2710 LAT2#=VAL(LEFT$(LAT$,2))+VAL(MID$(LAT$,4,2))/60:L#=VAL(MID$(LAT$,6,2-F)):IF F THEN L#=L#/600 ELSE L#=L#/3600
2720 LAT2#=(LAT2#+L#)*DR#
2730 LNG2#=VAL(LEFT$(LNG$,3))+VAL(MID$(LNG$,5,2))/60:L#=VAL(MID$(LNG$,7,2-F)):IF F THEN L#=L#/600 ELSE L#=L#/3600
2740 LNG2#=(LNG2#+L#)*DR#
2750 A=A+6+FLG*18:RETURN
2800 'BUILD FLIGHT DATA ***
2810 ID=VAL(MID$(FLT$,A,1)):GOSUB 900:IF ID<2 THEN GOSUB 2600:ID2$=I$:RETURN
2820 ID2$=MID$(FLT$,A+1,5):A=A+5:GOSUB 2600:GOSUB 3700:A=A+9:RETURN
2850 GOSUB 2600:GOSUB 3600:VAR1#=VAR2#:DIST=D#*RS#*10:R#=A#+VAR2#+PI#:IF R#>CR# THEN R#=R#-CR#
2860 RAD=R#*RD#*10:ID=4:GOTO 3880
2900 'BUILD FLT$ POINTERS ***
2910 ID=VAL(MID$(FLT$,A,1)):IF ID=5 THEN RETURN
2920 FLG=VAL(MID$(FLT$,A+5,1)):IF ID=4 THEN GOSUB 2750:GOTO 2910
2930 I=I+1:FLP$=FLP$+CHR$(A):IF ID=2 THEN GOSUB 2750:GOTO 2910
2940 IF ID<2 THEN 2600
2950 A=A+5:GOSUB 2600:GOSUB 3700:A=A+9:RETURN
```

```
3000 'FLIGHT POSITION ***
3010 IF ID=1 AND INDEX THEN GOSUB 6500:ID1$=A$
3020 IF INDEX THEN FX$=FX$+AF$+I$+"0":RETURN ELSE REC=LEN(INDEX$)/8
3030 GOSUB 10600:FLG1=0:GOSUB 6200:E=0:IF FLG1=0 THEN A=1:RETURN
3040 IF REC>999 THEN K$="N":GOTO 3070
3050 PRINT CHR$(27)"[6;2HFILE":X$="22":Y=1:GOSUB 10400:A$="File with the Nav Data @Y/NA ":GOSUB 10100:GOSUB 10500:
PRINT CHR$(27)"[0;"C$"m"CHR$(27)"[22;34H"CSR$;
3060 K$=FNCAP$(INPUT$(1)):IF K$=CHR$(27) THEN A=1:RETURN ELSE IF K$<>"N" AND K$<>"Y" THEN 3060
3070 X$="6":Y=0:GOSUB 10400:IF K$="Y" THEN FX$=FX$+AF$+I$+"0":GOTO 6300 ELSE FX$=FX$+ID$+"1"+LAT$+LNG$+VAR$:RETURN
3100 'INTERSECTION ***
3110 M$=FX$+LEFT$(ID1$+"     ",5):X$="8":Y=1:GOSUB 10400:A$=ID1$+" REFERENCE NAVAID":GOSUB 10100::ID=0:GOSUB 900:LS
ET AF$="0":ID2$=ID1$
3120 GOSUB 1400:IF A=1 THEN RETURN ELSE FX$="":GOSUB 3000:IF K$=CHR$(27) THEN RETURN ELSE IF A=1 THEN X$="6":Y=0:G
OSUB 10400:GOSUB 6350:GOTO 3120
3130 GOSUB 6350:X$="12":Y$="13":A$="Identifier: "+ID1$:GOSUB 10200
3140 X$="15":Y=20:Y$="13":A$="Radial:":GOSUB 10200:L=5:N=4:GOSUB 500:IF A=1 THEN IF K$=CHR$(27) THEN RETURN ELSE Y
=12:GOSUB 10400:GOTO 3120
3150 T!=VAL(A$):T!=(T!*10 MOD 3600)/10:IF T!=0 THEN T!=360
3160 A$=STR$(T!+1000):A$=RIGHT$(A$,LEN(A$)-2):IF T!=INT(T!) THEN A$=A$+".0"
3170 RAD$=LEFT$(A$+"   ",5):A$=A$+"x":GOSUB 590
3180 X$="18":Y$="13":Y=22:A$="Distance:":GOSUB 10200:L=4:N=4:GOSUB 500:IF A=1 THEN IF K$=CHR$(27) THEN RETURN ELSE
 Y=12:GOSUB 10400:GOTO 3140
3190 T!=VAL(A$):A$=STR$(T!):A$=RIGHT$(A$,LEN(A$)-1):IF T!=INT(T!) THEN A$=A$+".0"
3200 DIST$=LEFT$(A$+" ",4):A$=A$+" NM":GOSUB 590:X$="22":GOSUB 11300
3210 K$=FNCAP$(INPUT$(1)):IF K$=CHR$(27) THEN A=1:RETURN ELSE IF K$<>"N" AND K$<>"Y" THEN 3210
3220 IF K$="N" THEN Y=8:GOSUB 10400:GOTO 3180 ELSE FX$=M$+FX$+RAD$+DIST$:RETURN
3300 'TERMINAL WAYPOINT ***
3310 FAC$="":IF ID=1 THEN M$=ID2$+" TERMINAL WAYPOINT":GOSUB 1900:ID=0:LSET AF$="0":FX$="4":GOSUB 1400 ELSE RETURN
3320 IF A>1 THEN GOSUB 900:GOSUB 3000:IF A=1 THEN ID=1:GOTO 3310 ELSE FLT$=FLT$+FX$
3330 RETURN
3400 'DISPLAY FLIGHT ***
3410 C$=C$(2):B$=B$(2):H$="FLIGHT ROUTE":GOSUB 10000:A=1:D=9:FLG4=1:GOSUB 10550:GOSUB 20000:PRINT CHR$(27)"["C$";1
m"
3420 ID=VAL(MID$(FLT$,A,1)):FLG=VAL(MID$(FLT$,A+5,1)):GOSUB 900:IF ID<4 THEN GOSUB 3500:IF K$=CHR$(27) THEN RETURN
3430 ON ID GOTO 3440,3440,3460,3470,3480
3440 A$=MID$(FLT$,A+1,4):IF VAL(MID$(FLT$,A+5,1))=0 THEN INDEX=INSTR(INDEX$,A$):FLG5=1:GOSUB 6500
3450 GOSUB 10300:S$=" "+RIGHT$("    "+A$,4)+" "+FAC$:PRINT S$;:GOSUB 2750:GOTO 3550
3460 A$=MID$(FLT$,A+1,5):GOSUB 10300:A=A+5:PRINT RIGHT$("    "+A$,5)+" "+FAC$:GOSUB 2750:A=A+9:GOTO 3550
3470 PRINT " * "+FAC$+" "+MID$(FLT$,A+2,3):GOSUB 2750:GOTO 3420
3480 A$=FNID$(A$):GOSUB 10300:IDT$=A$:H$=IDF$+" to "+IDT$:GOSUB 10080:RD=1:X$="23":GOSUB 11300
3490 K$=FNCAP$(INPUT$(1)):IF K$<>CHR$(27) AND K$<>"N" AND K$<>"Y" THEN 3490 ELSE RETURN
3500 'ROUTE DISPLAY ROUTINE ***
3510 IF INKEY$=CHR$(27) THEN K$=CHR$(27):RETURN ELSE PRINT CHR$(27)"["RIGHT$(STR$(D+100),2)";6H";:D=D+1:IF D<22 TH
EN RETURN
3520 X$="23":GOSUB 11200:IF K$=CHR$(27) THEN RETURN
3530 GOSUB 10000:GOSUB 10550:PRINT CHR$(27)"["C$";1m"CHR$(27)"[9;6H"S$:D=10:GOTO 3510
3550 IF FLG4 THEN XX$=A$:A$=FNID$(A$):YY$=A$:GOSUB 10300:ZZ$=A$:IDF$=A$:FLG4=0
3560 GOTO 3420
3600 'CALCULATE DISTANCE ANGLE & COURSE ***
3610 D#=SIN(LAT1#)*SIN(LAT2#)+COS(LAT1#)*COS(LAT2#)*COS(LNG1#-LNG2#):IF D#>=1 THEN D#=0:A#=CR#:RETURN
3620 D#=FNACOS#(D#):IF FLG1 THEN RETURN
3630 A#=SIN(LAT1#)*SIN(LAT2#)+COS(LAT1#)*COS(LAT2#):IF A#>=1 THEN A#=0:GOTO 3650
3640 A#=FNACOS#(A#)
3650 A#=SIN(A#)/SIN(D#):IF A#=1 THEN A#=C#:GOTO 3670
3660 A#=FNASIN#(A#)
3670 A#=A#*SGN(LAT2#-LAT1#):IF LNG1#>LNG2# THEN A#=C#-A#:RETURN
3680 A#=A#+3*C#:RETURN
3700 'CALCULATE INTERSECTION COORDINATES ***
3710 A$=MID$(FLT$,A,5):GOSUB 10300:RAD$=A$:A$=MID$(FLT$,A+5,4):GOSUB 10300:DME$=A$
3720 R#=VAL(RAD$)*DR#-VAR2#:D#=VAL(DME$)/RS#:L#=LAT2#
3730 A#=SIN(D#)*COS(R#):LAT2#=LAT2#+FNASIN#(A#)
3740 A#=SIN(D#)*SIN(R#)/COS((L#+LAT2#)/2):LNG2#=LNG2#-FNASIN#(A#):RETURN
3800 'CALCULATE WAYPOINT ***
3810 GOSUB 2600:GOSUB 3600:B#=A#-TC#:IF ABS(B#)>PI# THEN B#=B#-CR#*SGN(B#)
3820 IF ABS(B#)>C# THEN RETURN
3830 L#=SIN(D#)*SIN(C#-ABS(B#)):L#=FNASIN#(L#):WD=L#*RS#:IF LL<=WD OR WD<=WL THEN RETURN
3840 LD=WD-WL:WL=WD:DIST$=STR$(LD):DREM$=STR$(S+LL-WL):GOSUB 4500:FLG2=2:GOSUB 3900:IF K$=CHR$(27) THEN RETURN
3850 R#=A#+2*C#+SGN(B#)*(C#-ABS(B#))+VAR2#:IF R#<0 THEN R#=R#+CR# ELSE IF R#>CR# THEN R#=R#-CR#
3860 RAD=R#*10*RD#:D#=SIN(D#)*SIN(ABS(B#)):DIST=FNASIN#(D#)*10*RS#:ID2$=I$
3870 VAR1#=VAR2#:FLG3=1:GOSUB 3880:FLG2=1:ID=2:GOSUB 900:GOTO 3900
3880 A$=STR$(RAD):GOSUB 3890:RAD$=A$:A$=STR$(DIST):GOSUB 3890:DME$=A$:RETURN
3890 A$=MID$(A$,2,LEN(A$)-2)+"."+RIGHT$(A$,1):RETURN
3900 'FLIGHT DISPLAY ROUTINE ***
3910 IF INKEY$=CHR$(27) THEN K$=CHR$(27):RETURN ELSE PRINT CHR$(27)"["RIGHT$(STR$(D+100),2)";1H"CHR$(27)"["C$";1m"
;
3915 D=D+1:IF D<22 THEN ON FLG2 GOTO 3930,3950,3980 ELSE IF P=0 THEN X$="23":GOSUB 11200:IF K$=CHR$(27) THEN RETUR
```

```
N
3920 FOR L=8 TO 20:X$=RIGHT$(STR$(L+100),2):Y=0:GOSUB 10400:NEXT:X$="23":GOSUB 10400:PRINT CHR$(27)"[8;1H"CHR$(27)
"["C$";1m"S$:D=9:GOTO 3910
3930 S$=ID2$+" "+FAC$:M$=S$:IF ID<2 THEN 3980 ELSE IF ID>2 THEN S$=S$+" * "+FNID$(I$):M$=S$
3940 S$=S$+" "+RAD$+D$+" "+DME$+" NM":M$=M$+" "+RAD$+PRDEG$+" "+DME$+" NM":GOTO 3980
3950 S$="  "+MC1$+D$:M$="  "+MC1$+PRDEG$:IF LEN(MC2$) THEN S$=S$+MC2$+D$:M$=M$+MC2$+PRDEG$
3960 A$="  DIST="+RIGHT$(DIST$,LEN(DIST$)-1):IF VAL(DREM$) THEN A$=A$+"  REM="+RIGHT$(DREM$,LEN(DREM$)-1)
3970 S$=S$+A$:M$=M$+A$
3980 IF P THEN LPRINT M$
3990 PRINT S$:RETURN
4000 'CALCULATE AND DISPLAY THE ROUTE ***
4010 GOSUB 10000:X$="6":Y$="14":A$="@ * STANDBY * A":GOSUB 10200:IF P THEN GOSUB 10540 ELSE GOSUB 10550
4015 GOSUB 20000:A=1:I=0:S=0:FLG1=1:FLP$="":GOSUB 2900
4020 LAT1#=LAT2#:LNG1#=LNG2#:VAR1#=VAR2#:GOSUB 2900:IF ID<5 THEN GOSUB 3600:S=S+D#*RS#:GOTO 4020
4030 FLG1=0:IF P THEN LPRINT PRINIT$ SPACE$((40-LEN(H$))/2) H$:ON ERROR GOTO 400
4040 M$=STR$(S)+" NM":S$=SPACE$(9)+"Total Distance:"+M$:M$=RIGHT$(M$,LEN(M$)-1):M$=SPACE$((40-LEN(M$))/2)+M$:FLG2=
3:D=6:GOSUB 3900:IF K$=CHR$(27) THEN 4400 ELSE IF P THEN LPRINT STRING$(40,61)
4050 D=8:A=1:IF RD<0 THEN A=ASC(RIGHT$(FLP$,1))
4100 GOSUB 2800:LAT1#=LAT2#:LNG1#=LNG2#:VAR1#=VAR2#:IF MID$(FLT$,A,1)="4" THEN GOSUB 2850
4110 FLG2=1:GOSUB 3900:IF K$=CHR$(27) THEN 4400 ELSE C=0:FOR LEG=(2*-(RD>0)+(I-1)*-(RD<0)) TO (I*-(RD>0)+-(RD<0))
STEP RD
4200 FLG3=0:A=ASC(MID$(FLP$,LEG,1)):IF MID$(FLT$,A,1)="2" THEN C=C+1:GOTO 4300
4210 GOSUB 2800:GOSUB 3600:LL=D#*RS#:S=S-LL:TC#=A#:WL=0:WHILE C:A=ASC(MID$(FLP$,LEG-C*RD,1)):C=C-1:GOSUB 3800:IF K
$=CHR$(27) THEN 4400 ELSE WEND
4220 IF FLG3 THEN A=ASC(MID$(FLP$,LEG,1)):GOSUB 2800
4230 GOSUB 4500:DIST$=STR$(LL-WL):DREM$=STR$(S):FLG2=2:GOSUB 3900:IF K$=CHR$(27) THEN 4400 ELSE LAT1#=LAT2#:LNG1#=
LNG2#:VAR1#=VAR2#:IF MID$(FLT$,A,1)="4" THEN GOSUB 2850
4240 FLG2=1:GOSUB 3900:IF K$=CHR$(27) THEN 4400
4300 NEXT LEG:IF P THEN FOR L=1 TO 9:LPRINT:NEXT
4310 PRINT CHR$(27)"[0;30;40m"CHR$(27)"[25;1H"SPC(39)CHR$(27)"[1A":X$="23":GOSUB 11200
4400 CLOSE 1:GOTO 2400
4500 'FORMAT & PRINT MC
4510 MC#=TC#+VAR1#:GOSUB 4550:MC1$="MC="+A$:MC2$="":IF VAR1#<>VAR2# THEN MC#=TC#+VAR2#:GOSUB 4550:MC2$="/"+A$
4520 RETURN
4550 IF MC#<0 THEN MC#=MC#+CR# ELSE IF MC#>CR# THEN MC#=MC#-CR#
4560 A$=RIGHT$(STR$(CINT(MC#*RD#+1000)),3):RETURN
5000 'LOAD FLIGHT ROUTE ***
5010 C$=C$(2):B$=B$(2):H$="LOAD FLIGHT DATA":GOSUB 10000:C$=C$+";1":A$="":GOSUB 10580:GOSUB 5400:IF K$=CHR$(27) TH
EN 300
5020 PRINT CHR$(27)"[22;11H"CHR$(27)"[0;37mLoad which flight? "CSR$;
5030 K$=FNCAP$(INPUT$(1)):IF K$=CHR$(27) OR K$=CHR$(13) THEN 300 ELSE IF K$<"A" OR K$>"Z" THEN 5030
5040 PRINT K$:OPEN "FLTDATA."+K$ FOR INPUT AS 2:INPUT #2,ID1$
5050 IF ID1$="EMPTY" THEN X$="22":Y=29:M$="* EMPTY *":GOSUB 11080:CLOSE 2:GOTO 5020
5060 INPUT #2,ID2$,FLT$:CLOSE 2:GOSUB 3400:CLOSE 1:IF K$=CHR$(27) THEN 300 ELSE IF K$="N" THEN 5000 ELSE 2400
5400 'FLIGHT FILE INDEX ***
5410 ON ERROR GOTO 5450:FOR J=0 TO 1:Y$=RIGHT$(STR$(J*20+102),2):FOR I=7 TO 19:X$=RIGHT$(STR$(I+100),2):K$=CHR$(13
*J+I+58)
5420 OPEN "FLTDATA."+K$ FOR INPUT AS 2:INPUT #2,IDS$:A$=IDS$:IF IDS$<>"EMPTY" THEN INPUT#2,IDS$:A$=A$+" to "+IDS$
5430 CLOSE 2:A$="@]"+K$+"^A "+A$:GOSUB 10200:IF INKEY$=CHR$(27) THEN I=19:J=1:K$=CHR$(27)
5440 NEXT:NEXT:ON ERROR GOTO 400:RETURN
5450 IF ERR=53 THEN OPEN "FLTDATA."+K$ FOR OUTPUT AS 2:WRITE #2,"EMPTY":CLOSE 2:RESUME ELSE GOTO 400
5500 'SAVE FLIGHT ROUTE ***
5510 C$=C$(2):B$=B$(2):ID1$=IDF$:ID2$=IDT$:IF RD<0 THEN SWAP ID1$,ID2$
5520 H$="SAVE FLIGHT "+ID1$+" to "+ID2$:GOSUB 10000:A$="":GOSUB 10580:C$=C$+";1":GOSUB 5400:IF K$=CHR$(27) THEN 24
00
5530 PRINT CHR$(27)"[22;8H"CHR$(27)"[0;37mStore in which position? "CSR$;
5540 K$=FNCAP$(INPUT$(1)):IF K$=CHR$(27) OR K$=CHR$(13) THEN 2400 ELSE IF K$<"A" OR K$>"Z" THEN 5540
5550 PRINT K$:OPEN "FLTDATA."+K$ FOR OUTPUT AS 2:WRITE #2,ID1$,ID2$,FLT$:CLOSE 2:GOTO 2400
5900 'NAV DATA FULL MESSAGE ***
5910 PRINT CHR$(27)"["C$(0)"m"CHR$(27)"[17;10HNAV DATA FILE IS FULL!"
5920 PRINT CHR$(7):T=2000:WHILE T:T=T-1:WEND:GOSUB 6400
6000 FLG5=0:C$=C$(3):H$="NAV DATA MENU":GOSUB 10000:X$="9":Y$="14":B$=B$(2):C$=C$(2)+";1":A$="@ 1 A  ADD/EDIT":GOS
UB 10200
6010 X$="11":B$=B$(3):C$=C$(3)+";1":A$="@ 2 A  DELETE":GOSUB 10200
6020 X$="13":B$=B$(5):C$=C$(5)+";1":A$="@ 3 A  LIST":GOSUB 10200
6030 X$="15":B$=B$(4):C$=C$(4)+";1":A$="@ 4 A  PRINT":GOSUB 10200
6040 X$="17":B$=B$(1):C$=C$(1)+";1":A$="@ 5 A  FORMAT":GOSUB 10200
6050 X$="20":B$=B$(0):C$=C$(0)+";1":A$="@ 0 A  MAIN MENU":GOSUB 10200:PRINT MENU$;
6060 K$=INPUT$(1):IF K$<"0" OR K$>"5" THEN 6060 ELSE K=VAL(K$):IF K=0 THEN 300
6070 IF K=5 THEN GOSUB 9000:GOTO 6000 ELSE ON K GOTO 6100,6600,7700,7000
6100 C$=C$(2):CS$=C$+";1":H$="ADD/EDIT NAV DATA":GOSUB 10000:GOSUB 20000:FLG=1:GOSUB 10600
6120 E2=0:GET 1,1:REC=CVI(REC$):IF REC>999 THEN E2=1
6130 PRINT CHR$(27)"["C$"m"CHR$(27)"[6;1H"REC"Entries   ":GOSUB 10800
6140 FLG1=0:FLG2=0:GOSUB 10700:IF I THEN GOSUB 6400:GOTO 6000 ELSE GOSUB 10500
6170 PRINT CHR$(27)"[6;2H"CHR$(27)"["C$"m"FAC$"      ":E=0:E1=0:S$="":GOSUB 1400:IF A=1 THEN 6330
6180 IF INDEX THEN GOSUB 6500:GOSUB 6220:GOTO 6330 ELSE IF E2 THEN 5900 ELSE GOSUB 6200:GOTO 6330
```

```
6200 PRINT CHR$(27)"[6;2HADD "FAC$:FOR I=1 TO 3:ON I GOSUB 1500,1600,1700:IF A=1 THEN RETURN
6210 NEXT I:FLG1=1
6220 GOSUB 10520:PRINT CHR$(27)"[6;2H"CHR$(27)"["C$"mEDIT "FAC$:C$=CS$
6230 GOSUB 10900
6240 K$=INPUT$(1):IF K$=CHR$(13) THEN A=2:RETURN ELSE IF K$=CHR$(27) THEN FLG1=0:RETURN
6250 IF K$<"1" OR K$>"4" THEN 6240 ELSE E=1:X$="22":Y=0:GOSUB 10400:ON VAL(K$) GOSUB 6260,6270,6280,6290:GOTO 6230
6260 S$=ID$:GOTO 1400
6270 S$=LAT$:GOTO 1500
6280 S$=LNG$:GOTO 1600
6290 S$=VAR$:GOTO 1700
6300 'STORE NAV DATA
6310 IF FLG1 THEN PUT 1,REC+2:FLG=0:IF E1=0 THEN INDEX$=INDEX$+AF$+I$+RIGHT$("00"+RIGHT$(STR$(REC),LEN(STR$(REC))-
1),3)+"~":GET 1,1:REC=REC+1:LSET REC$=MKI$(REC):PUT 1,1
6320 RETURN
6330 GOSUB 6300:GOSUB 6350:GOTO 6120
6350 X$="12":Y=0:GOSUB 10400:X$="14":GOSUB 10400:X$="15":GOSUB 10400:X$="16":GOSUB 10400:X$="18":GOSUB 10400:X$="2
2":GOSUB 10400
6360 PRINT CHR$(27)"[u":RETURN
6400 'SORT ***
6410 CLOSE 1:IF FLG THEN RETURN ELSE C$=C$(2):H$="SORTING THE NAV DATA":IF S THEN GOSUB 10080:S=0 ELSE GOSUB 10000
:GOSUB 11100
6415 A$=" sorting":GOSUB 10580
6420 FLG=0:FOR T=0 TO LEN(INDEX$)/8-2:X=T*8+1:Y=X+8
6430 IF MID$(INDEX$,X+1,3)=MID$(INDEX$,Y+1,3) AND MID$(INDEX$,X,1)<MID$(INDEX$,Y,1) THEN 6470
6440 IF MID$(INDEX$,X+1,3)<MID$(INDEX$,Y+1,3) AND (ASC(MID$(INDEX$,X+1,1))>64 OR ASC(MID$(INDEX$,Y+1,1))<65) THEN
6470
6450 IF ASC(MID$(INDEX$,X+1,1))>64  AND ASC(MID$(INDEX$,Y+1,1))<65 THEN 6470
6460 S$=MID$(INDEX$,X,8):MID$(INDEX$,X,8)=MID$(INDEX$,Y,8):MID$(INDEX$,Y,8)=S$:FLG=1
6470 IF INKEY$=CHR$(27) THEN T=1000:FLG=0
6480 NEXT T:IF FLG THEN 6420 ELSE OPEN "NAVDATA.NDX" FOR OUTPUT AS 2:WRITE #2,INDEX$:CLOSE 2:RETURN
6500 'GET RECORD ***
6510 REC=VAL(MID$(INDEX$,INDEX+4,3)):GET 1,REC+2:RECNDX=INDEX+1:A$=FNID$(ID$):IF FLG5 THEN A$=ID$:RETURN
6520 GOSUB 590:E1=1:GOSUB 1570:X$="14":Y$="7":A$="@ 2 A Latitude: "+DEG$:GOSUB 10200
6530 GOSUB 1670:X$="16":A$="@ 3 A Longitude: "+DEG$:GOSUB 10200
6540 VAR=VAL(VAR$):GOSUB 1750:X$="18":A$="@ 4 A Magnetic Variation: "+DEG$:GOTO 10200
6600 'DELETE NAV DATA ***
6610 C$=C$(0):H$="DELETE NAV DATA":GOSUB 10000:GOSUB 20000:FLG=1
6620 GOSUB 10800:GOSUB 10700:IF I THEN GOSUB 6800:GOTO 6000 ELSE GOSUB 10530
6650 PRINT CHR$(27)"[8;14H"CHR$(27)"["C$(0)"mDELETE "FAC$:E=0:E1=0:S$=""
6660 GOSUB 1400:IF A=1 THEN 6690 ELSE IF INDEX THEN GOSUB 6500 ELSE M$="* NOT FOUND *":GOSUB 11080:GOTO 6660
6670 PRINT CHR$(27)"[22;2H"CHR$(27)"[1;"C$(0)"mAre you sure you want to DELETE Y/N? "CSR$;
6680 K$=FNCAP$(INPUT$(1)):IF K$=CHR$(27) OR K$=CHR$(13) OR K$="N" THEN 6690 ELSE IF K$<>"Y" THEN 6680 ELSE LSET AF
$="X":PUT 1,REC+2:FLG=0:MID$(INDEX$,INDEX,1)=AF$
6690 X$="8":Y=2:GOSUB 10400:GOSUB 6350:X$="22":GOSUB 10400:GOTO 6620
6800 'PACK & REINDEX ***
6810 IF FLG THEN CLOSE 1:RETURN ELSE C$=C$(0):H$="PACK & REORGANIZE NAV DATA":GOSUB 10000:GOSUB 11100:S=1
6820 INDEX$="":OPEN "TEMP" AS 2 LEN=23:FIELD 2,23 AS DAT$:FIELD 1,23 AS OLD$:REC=0
6830 GET 1,1:FOR X=1 TO CVI(REC$):GET 1,X+1:IF AF$<>"X" THEN LSET DAT$=OLD$:PUT 2,REC+2:INDEX$=INDEX$+AF$+RIGHT$(I
D$,3)+RIGHT$("00"+RIGHT$(STR$(REC),LEN(STR$(REC))-1),3)+"~":REC=REC+1
6840 NEXT X:GET 2,1:LSET DAT$=MKI$(REC):PUT 2,1:CLOSE 1,2:KILL "NAV.DAT":NAME "TEMP" AS "NAV.DAT":GOTO 6400
6960 Y=1:GOSUB 10400:GOSUB 10500:LSET AF$=RIGHT$(STR$(ID),1):GOSUB 900:PRINT CHR$(27)"[6;2H"FAC$"       "
7000 'PRINT NAV DATA ***
7010 C$=C$(4):H$="PRINT NAV DATA":GOSUB 10000:GOSUB 20000
7020 X$="6":S$=STR$(REC)+" Entries ":A$=S$:GOSUB 10100:PRINT CHR$(27)"[9;2HPrepare Printer {top-of-formzzon-line}"
7030 M$="PRINT":B$=B$(4):GOSUB 7600:IF FLG THEN 7450 ELSE GOSUB 10000:IF ID<0 THEN X$="6":A$=S$:GOSUB 10100
7100 'PRINTING ***
7120 GOSUB 10540:PRINT CHR$(27)"[15;17H"CHR$(27)"[1;"C$(0)";5mPRINTING"
7130 S$=FAC$+" DATA":D$=PRDEG$:I=0:PAGE=1
7200 'HEADING
7210 ON ERROR GOTO 7500:LPRINT PRINIT$ SPACE$((80-LEN(S$))/2) S$:ON ERROR GOTO 400
7220 LPRINT TAB(9) "Printed: " MID$("  JanFebMarAprMayJunJulAugSepOctNovDec",VAL(LEFT$(DATE$,2))*3,3) " " MID$(DAT
E$,4,2) ", " RIGHT$(DATE$,4) TAB(60) "Page ";
7230 LPRINT USING "##";PAGE;:LPRINT " of ";:LPRINT USING "##";INT(REC/50+.99)
7235 LPRINT TAB(9) STRING$(64,"="):T=2*F
7240 LPRINT TAB(13) "FACILITY" TAB(30) "VARIATION" TAB(45+T) "LATITUDE" TAB(62+T) "LONGITUDE":LPRINT TAB(9) STRING
$(64,"-"):LPRINT:P=1
7300 'PRINT ***
7310 IF INKEY$=CHR$(27) THEN 7410 ELSE GET 1,VAL(MID$(INDEX$,I*8+5,3))+2:I=I+1
7320 IF AF<0 OR AF=VAL(AF$) THEN ID=VAL(AF$):GOSUB 900 ELSE 7400
7330 IF AF<0 THEN LPRINT TAB(9)ID$ "  -  " FAC$; ELSE LPRINT TAB(15)ID$;
7340 VAR=VAL(VAR$):GOSUB 1750:LPRINT TAB(32)RIGHT$(" "+DEG$,24);
7350 GOSUB 1570:LPRINT SPC(7+T)DEG$;
7360 GOSUB 1670:LPRINT SPC(6+T)DEG$:P=P+1
7400 IF REC>I THEN IF P>50 THEN PAGE=PAGE+1:LPRINT CHR$(12):GOTO 7200 ELSE 7300
7410 LPRINT CHR$(12):D$="x"
7450 CLOSE 1:GOTO 6000
```

```
7500 'PRINTER ERROR ***
7510 GOSUB 7550:RESUME 7450
7520 GOSUB 7550:CLOSE 1:RESUME 2400
7550 C$=C$(0):GOSUB 10000:PRINT CHR$(27)"[15;14HPRINTER ERROR!"CHR$(7):X$="20":ON ERROR GOTO 400:GOTO 11200
7600 'DATA CHOICE ***
7610 C$=C$+";1":FLG=0:X$="13":Y$="9":A$="@ RETURN A All Nav Data":GOSUB 10200:X$="15":Y$="14":A$="@ A A Airports":
GOSUB 10200
7620 X$="17":A$="@ N A Navaids":GOSUB 10200:A$="":GOSUB 10580:PRINT CHR$(27)"[0;37m"CHR$(27)"[21;10H"M$" which Nav
 Data? "CSR$;
7630 K$=FNCAP$(INPUT$(1)):IF K$=CHR$(27) OR REC=0 THEN FLG=1:RETURN ELSE IF K$<>CHR$(13) AND K$<>"A" AND K$<>"N" T
HEN 7630
7640 ID=0:IF K$="A" THEN ID=1 ELSE IF K$<>"N" THEN ID=-1
7650 GOSUB 900:AF=ID:IF ID<0 THEN FAC$="NAV"
7660 H$=M$+" "+FAC$+" DATA":C$=LEFT$(C$,2):RETURN
7700 'LIST NAV DATA ***
7710 C$=C$(5):H$="LIST NAV DATA":GOSUB 10000:GOSUB 20000
7720 X$="6":S$=STR$(REC)+" Entries ":A$=S$:GOSUB 10100:M$="LIST":B$=B$(5):GOSUB 7600:IF FLG THEN 7890 ELSE I=0:FLG
=1
7800 'LIST ***
7810 IF INKEY$=CHR$(27) THEN 7890 ELSE GET 1,VAL(MID$(INDEX$,I*8+5,3))+2:I=I+1
7820 IF AF<0 OR AF=VAL(AF$) THEN ID=VAL(AF$):GOSUB 900 ELSE IF I>=REC THEN 7870 ELSE 7810
7830 IF FLG THEN GOSUB 10000:GOSUB 10550:D=9:X$="9":Y$="10":FLG=0
7840 S$=CHR$(27)+"["+C$+";1m"+ID$:IF AF<0 THEN S$=S$+CHR$(27)+"[0;"+C$+"m - "+FAC$
7850 GOSUB 1570:PRINT CHR$(27)"["X$";1H"S$CHR$(27)"[0;"C$"m"CHR$(27)"["X$";20HLatitude  "CHR$(27)"[1m"DEG$
7860 VAR=VAL(VAR$):GOSUB 1750:S$=CHR$(27)+"[0;"+C$+"mMag Var "+CHR$(27)+"[1m"+DEG$:GOSUB 1670:PRINT CHR$(27)"["Y$"
;2H"S$CHR$(27)"[0;"C$"m"CHR$(27)"["Y$";18HLongtitude "CHR$(27)"[1m"DEG$
7870 X$="25":IF I>=REC THEN IF FLG THEN 7890 ELSE GOSUB 11210:GOTO 7890
7880 IF D<21 THEN D=D+3:X$=RIGHT$(STR$(D),2):Y$=RIGHT$(STR$(D+1),2):GOTO 7800 ELSE GOSUB 11210:IF K$=CHR$(27) THEN
 7890 ELSE C$=C$(5):FLG=1:GOTO 7800
7890 CLOSE 1:GOTO 6000
8000 'PRINTER CODES ***
8010 C$=C$(4):H$="PRINTER CODES":GOSUB 10000:GOSUB 10520
8020 C$=C$(4):B$=B$(4):X$="9":Y$="2":A$="@ 1 A  Printer Initialization y y y y y y":GOSUB 10200
8030 PRINT CHR$(27)"[11;1H"CHR$(27)"[1m";:FOR T=1 TO 10:PRINT FNZERO$(MID$(PRINIT$,T,1));:NEXT T
8040 X$="15":A$="@ 2 A  Degree Symbol Printer Codes  y y y":GOSUB 10200
8050 PRINT CHR$(27)"[17;1H"CHR$(27)"[1m";:FOR T=1 TO 20:IF T=11 THEN PRINT CHR$(27)"[19;1H";
8060 PRINT FNZERO$(MID$(PRDEG$,T,1));:NEXT T:FLG2=1
8070 C$=C$(4):GOSUB 10900
8080 K$=INPUT$(1):IF K$=CHR$(27) OR K$=CHR$(13) THEN 300 ELSE IF K$<"1" OR K$>"2" THEN 8080
8100 X$="22":Y=1:GOSUB 10400:C$=C$(4):A$="ENTER DECIMAL ASCII CODES":GOSUB 10100:K=VAL(K$):IF K=1 THEN PR$=PRINIT$
 ELSE PR$=PRDEG$
8110 L=3:N=1:FOR T=1 TO 10*K:X$=RIGHT$(STR$(111+(K-1)*6-(T>10)*2),2):Y=1+(T-1)*4+(T>10)*40:S$=RIGHT$(FNZERO$(MID$(
PR$,T,1)),3)
8120 A=1:D=0:C$=C$+";1":GOSUB 500:IF K$=CHR$(27) THEN 300
8130 S$=CHR$(VAL(A$) MOD 256):Y=Y-1:A$=FNZERO$(S$):GOSUB 590:MID$(PR$,T,1)=S$:IF A=1 THEN 8140 ELSE NEXT T
8140 GOSUB 20200:IF K=1 THEN PRINIT$=PR$:LSET PR1$=PR$ ELSE PRDEG$=PR$:LSET PR2$=PR$
8150 PUT 3,1:CLOSE 3:GOTO 8070
8500 'INITIALIZE THE SCREEN ***
8510 C$="37":B$=";47m":H$="SCREEN INITIALIZATION":GOSUB 10000:C$="37;1"
8520 X$="12":Y$="10":A$="@ 1 A  MONOCHROME":GOSUB 10200
8530 X$="14":A$="@ 2 A  COLOR":GOSUB 10200
8540 PRINT CHR$(27)"[0;37m"CHR$(27)"[22;6HWhich monitor do you have? "CSR$;
8560 K$=INPUT$(1):IF K$=CHR$(13) OR K$=CHR$(27) THEN 300 ELSE IF K$<"1" OR K$>"2" THEN 8560
8570 GOSUB 20200:CFLG=VAL(K$)-1:LSET CFLG$=RIGHT$(STR$(CFLG),1):PRINT CHR$(27)"[="CFLG$"h"CHR$(27)"[0;30;40m"CHR$(
27)"[2J":C$="777777":IF CFLG THEN C$="143526"
8580 PUT 3,1:CLOSE 3:FOR I=0 TO 5:A$=MID$(C$,I+1,1):C$(I)="3"+A$:B$(I)=";4"+A$+"m":NEXT:GOTO 300
9000 'LAT/LNG FORMAT ***
9010 C$=C$(3):H$="LAT/LNG FORMAT":GOSUB 10000:GOSUB 20000:PRINT CHR$(27)"[6;1H"REC"Entries"
9020 GOSUB 10600:X$="9":Y$="5":C$=C$+";1":B$=B$(3):A$="@ 0 A  RETURN TO MENU":GOSUB 10200
9030 X$="12":A$="@ 1 A  DEGREES, MINUTES, & SECONDS":GOSUB 10200
9040 PRINT CHR$(27)"[14;10HFormat:  DDx MM' SS"CHR$(34)CHR$(27)"[15;15HExample:  90x 38' 47"CHR$(34)
9050 PRINT CHR$(27)"[16;10HEntry form:  DD.MMDD"CHR$(27)"[17;15HExample:  90.3847"
9060 X$="20":A$="@ 2 A  DEGREES, MINUTES, & TENTHS":GOSUB 10200
9070 PRINT CHR$(27)"[22;10HFormat:  DDx MM.d"CHR$(27)"[23;15HExample:  90x 38.8'"
9080 PRINT CHR$(27)"[24;10HEntry form:  DD.MMd"CHR$(27)"[25;15HExample:  90.388"CHR$(27)"[1A"
9100 FLG=1:K$=INPUT$(1):IF K$=CHR$(27) OR K$=CHR$(13) OR K$="0" THEN 9190 ELSE IF K$<"1" OR K$>"2" THEN 9100
9130 K=VAL(K$)-1:X$="9":Y=1:GOSUB 10400:FOR N=20 TO 25:PRINT CHR$(27)"["RIGHT$(STR$(N-K*8),2)";2H"CHR$(27)"[0;30;4
0m"CHR$(27)"[K";:NEXT
9140 X$=RIGHT$(STR$(20-K*4),2):PRINT CHR$(27)"["C$"m"CHR$(27)"["X$";2HYour choice is format option #"K$" Y/N?"CSR$
;
9150 K$=FNCAP$(INPUT$(1)):IF K$="N" OR K$=CHR$(27) THEN GOSUB 10400:GOTO 9020 ELSE IF K$<>"Y" AND K$<>CHR$(13) THE
N 9150
9180 IF K=F THEN 9190 ELSE FLG=0:F=K:GOSUB 20200:LSET F$=RIGHT$(STR$(K),1):PUT 3,1:CLOSE 3
9190 IF REC=0 OR FLG THEN 9260
9200 'REFORMAT LAT/LNG ***
9210 H$="REFORMATTING LAT/LNG":GOSUB 10000:PRINT CHR$(27)"[6;1H"REC"Entries":GOSUB 10600:GOSUB 11100
```

```
9220 FOR REC=1 TO REC:GET 1,REC+1:A$=LAT$:B$=LNG$:A=VAL(MID$(A$,6,2)):D=VAL(MID$(B$,7,2))
9230 IF F THEN GOSUB 9300 ELSE S$=STR$(A*6):MID$(A$,6,2)=RIGHT$("0"+RIGHT$(S$,LEN(S$)-1),2):S$=STR$(D*6):MID$(B$,7
,2)=RIGHT$("0"+RIGHT$(S$,LEN(S$)-1),2)
9250 LSET LAT$=A$:LSET LNG$=B$:PUT 1,REC+1:NEXT REC
9260 CLOSE 1:RETURN
9300 T=A/6:MID$(A$,6,2)=RIGHT$(STR$(T),1)+" "
9310 IF T=10 THEN A=VAL(MID$(A$,4,2)):T=A+1 ELSE 9400
9320 IF T=60 THEN S$=" 0" ELSE S$=STR$(T)
9330 MID$(A$,4,2)=RIGHT$("0"+RIGHT$(S$,LEN(S$)-1),2)
9340 IF T=60 THEN A=VAL(A$):A=A+1:S$=STR$(A):MID$(A$,1,2)=RIGHT$(" "+RIGHT$(S$,LEN(S$)-1),2)
9400 T=D/6:MID$(B$,7,2)=RIGHT$(STR$(T),1)+" "
9410 IF T=10 THEN D=VAL(MID$(B$,5,2)):T=D+1 ELSE RETURN
9420 IF T=60 THEN S$=" 0" ELSE S$=STR$(T)
9430 MID$(B$,5,2)=RIGHT$("0"+RIGHT$(S$,LEN(S$)-1),2)
9440 IF T=60 THEN D=VAL(B$):D=D+1:S$=STR$(D):MID$(B$,1,3)=RIGHT$("  "+RIGHT$(S$,LEN(S$)-1),3)
9450 RETURN
10000 'HEADING ***
10010 PRINT CHR$(27)"[0;30;40m":PRINT CHR$(27)"[2J"CHR$(27)"[1;10H"CHR$(27)"[1;37mAIR NAVIGATION/*RNAV*"
10020 PRINT CHR$(27)"[3;1H"CHR$(27)"[0;30;40m"SPC((39-LEN(H$))/2)CHR$(27)"[37;1m"H$
10030 PRINT CHR$(27)"[4;1H"STRING$(40,95):RETURN
10080 I=(39-LEN(H$))/2:PRINT CHR$(27)"[s"CHR$(27)"[3;1H"CHR$(27)"[0;30;40m"SPC(I)CHR$(27)"[37;1m"H$CHR$(27)"[0;30m
"SPC(I)CHR$(27)"[u";:RETURN
10100 'SCREEN ROUTINES ***
10110 Y=(40-LEN(A$))/2:Y$=RIGHT$(STR$(Y+101),2)
10200 PRINT CHR$(27)"[0;30;40m"CHR$(27)"["X$";"Y$"H"CHR$(27)"["C$"m";
10210 L=LEN(A$):R=INSTR(A$,"@"):N=INSTR(A$,"A"):IF R>0 THEN PRINT MID$(A$,1,R-1)CHR$(27)"[0;7"B$;
10220 IF N>0 THEN PRINT MID$(A$,R+1,N-R-1)CHR$(27)"[0;"C$"m"RIGHT$(A$,L-N):RETURN
10230 PRINT RIGHT$(A$,L-R):RETURN
10300 'TRUNCATE BLANKS ***
10310 L=LEN(A$):WHILE L:IF MID$(A$,L,1)<>" " THEN A$=LEFT$(A$,L):RETURN ELSE L=L-1:WEND:A$="":RETURN
10400 'ERASE LINE ***
10410 A$=CHR$(27)+"[0;30;40m"+CHR$(27)+"[K":GOTO 590
10500 'ESCAPE NOTATION ***
10510 A$=" entry":GOTO 10580
10520 A$=" change":GOTO 10580
10530 A$=" delete":GOTO 10580
10540 A$=" printing":GOTO 10580
10550 A$=" listing"
10580 PRINT CHR$(27)"[0;30;40m"CHR$(27)"[25;1H"CHR$(27)"[K"CHR$(27)"[25;"RIGHT$(STR$(INT(114-LEN(A$)/2)),2)"H"CHR$
(27)"[30"B$(0)" Esc "CHR$(27)"[40;"C$(0)"m to abort"A$CHR$(27)"[2A":RETURN
10600 'FORMAT DISPLAY ***
10610 A$="Format: DDx MM":IF F THEN A$=A$+".d'" ELSE A$=A$+"' SS"+CHR$(34)
10620 PRINT CHR$(27)"[0;1;"C$"m"CHR$(27)"[6;"RIGHT$(STR$(121+F*2),2)"H"A$:RETURN
10700 'A/N? ***
10710 X$="21":Y$="8":B$=B$(2):C$=C$(2)+";1":A$="AIRPORT or NAVAID? @ A/N A":GOSUB 10200:PRINT CHR$(27)"[21;33H"CSR
$;
10720 K$=FNCAP$(INPUT$(1)):IF K$=CHR$(27) OR K$=CHR$(13) THEN I=1:RETURN ELSE ID=0:IF K$="A" THEN ID=1 ELSE IF K$<
>"N" THEN 10720
10730 Y=1:GOSUB 10400:LSET AF$=RIGHT$(STR$(ID),1):I=0:GOTO 900
10800 'ESC/RETURN ***
10810 PRINT CHR$(27)"[25;8H"CHR$(27)"[0;30"B$(0)" Esc "CHR$(27)"[40;"C$(0)"m or "CHR$(27)"[30"B$(0)" Return "CHR$(
27)"[40;"C$(0)"m to Quit"CHR$(27)"[1A":RETURN
10900 'FIELD CHOICE ***
10910 A$="Enter Field to Changez@RETURNA to Accept":X$="22":Y$="2":GOTO 10200
11000 'INVALID/DUPLICATE MESSAGE ***
11010 M$="* INVALID *":GOTO 11080
11050 M$="* DUPLICATE *"
11080 M$=CHR$(27)+"[30"+B$(0)+M$+CHR$(27)+"[40m"
11090 GOSUB 10400:A$=M$:GOSUB 590:PRINT CHR$(7):T=1000:WHILE T:T=T-1:WEND:GOTO 10400
11100 'DO NOT DISTURB ***
11140 PRINT CHR$(27)"[1;5m"CHR$(27)"[15;14HDO NOT DISTURB"
11150 PRINT CHR$(27)"[0;1;"C$"m"CHR$(27)"[21;7HThis may take a short time.":RETURN
11200 'CONTINUE ***
11210 PRINT CHR$(27)"["X$";6H"CHR$(27)"[0;37mPress any key to continue-/ "CSR$;:K$=INPUT$(1):RETURN
11300 'CORRECT? ***
11310 Y$="9":A$="Is this correct? @ Y/N A ":GOSUB 10200:PRINT CHR$(27)"["X$";32H"CSR$;;:RETURN
20000 OPEN "NAV.DAT" AS 1 LEN=23:FIELD 1,1 AS AF$,4 AS ID$,7 AS LAT$,8 AS LNG$,3 AS VAR$
20010 FIELD 1,2 AS REC$:GET 1,1:REC=CVI(REC$):IF REC=LEN(INDEX$)/8 THEN RETURN
20100 C$=C$(0):H$="FILE-INDEX ERROR":GOSUB 10000
20110 PRINT CHR$(27)"[13;6HThe NAV DATA file needs to be"CHR$(27)"[15;15HRe-indexed!":A$="":GOSUB 10580
20120 X$="20":GOSUB 11200:IF K$=CHR$(27) THEN 400 ELSE GOSUB 6800:RUN
20200 OPEN "SYSTEM.DAT" AS 3:FIELD 3,1 AS CFLG$,1 AS F$,10 AS PR1$,20 AS PR2$:IF LOF(3)=0 THEN 20210 ELSE GET 3,1:
RETURN
20210 FIELD 3,128 AS A$:LSET A$="00"+STRING$(126,0):PUT 3,1:CLOSE 3:GOTO 20200
20300 IF ERR=53 THEN OPEN "NAVDATA.NDX" FOR OUTPUT AS 2:PRINT #2,"":CLOSE 2:RESUME ELSE GOTO 400
30000 'INTRO SCREEN ***
```

```
30010 PRINT CHR$(27)"[0;"C$(1)B$(1)CHR$(27)"[2J"CHR$(27)"[2;20H"CHR$(27)"[1;37m"CHR$(15)
30020 PRINT CHR$(27)"[3;17H"CHR$(27)"["C$(5)"mIMKOKM;"
30030 PRINT CHR$(27)"[4;17H"CHR$(27)"["C$(5)"m: : : :"
30040 PRINT CHR$(27)"[5;8H"CHR$(27)"[30m."CHR$(27)"[5;17H"CHR$(27)"["C$(5)"mHKJMJK<"
30050 PRINT CHR$(27)"[6;6H"CHR$(27)"[30mDMmMD"CHR$(27)"[6;18H"CHR$(27)"["C$(5)"m:    :"
30060 PRINT CHR$(27)"[7;18H:    :        "CHR$(27)"["C$(0)"m^["CHR$(16)
30070 PRINT CHR$(27)"[8;18H"CHR$(27)"["C$(5)"m:    :        3"
30080 PRINT CHR$(27)"[9;18H"CHR$(27)"["C$(5)"m:    :        3"
30090 PRINT CHR$(27)"[10;8H"CHR$(27)"["C$(5)"mIMMMMMMMMMJMMMJMMMMMMMOM;"
30100 PRINT CHR$(27)"[11;2HIMMMMM<"CHR$(27)"["C$(3)"m AIR NAVIGATION/*RNAV* "CHR$(27)"["C$(5)"mHMMMMMM;"
30110 PRINT CHR$(27)"[12;2H:IMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM;:"
30120 PRINT CHR$(27)"[13;2H::"CHR$(27)"["C$(2)"m        by Robert A. Tims          "CHR$(27)"["C$(5)"m::"
30130 PRINT CHR$(27)"[14;2H::                                        ::"
30140 PRINT CHR$(27)"[15;2H::                  "CHR$(27)"[37mB                "CHR$(27)"["C$(5)"m::"
30150 PRINT CHR$(27)"[16;2H::                  "CHR$(27)"[37m3                "CHR$(27)"["C$(5)"m::"
30160 PRINT CHR$(27)"[17;2H::                "CHR$(27)"[37mUO8                "CHR$(27)"["C$(5)"m::"
30170 PRINT CHR$(27)"[18;2H::      "CHR$(27)"[37m@DDDmDmDKMT"CHR$(27)"[30mi"CHR$(27)"[37m>MKDmDmDDDY     "CHR$(27)
"["C$(5)"m::"
30180 PRINT CHR$(27)"[19;2H::              "CHR$(27)"[30m{     {              "CHR$(27)"["C$(5)"m::"
30190 PRINT CHR$(27)"[20;1H"CHR$(27)"["C$(4)B$(4)"000"CHR$(27)"[30;47m22222222222222222222222222222222"CHR$(27)"
["C$(4)B$(4)"000"
30200 PRINT CHR$(27)"[21;1H00"CHR$(27)"[30;47m22222222222222222222222222222222"CHR$(27)"["C$(4)B$(4)"00"
30210 PRINT CHR$(27)"[22;1H0"CHR$(27)"[30;47m22222222222222222222222222222222"CHR$(27)"["C$(4)B$(4)"0"
30220 PRINT CHR$(27)"[23;1H"CHR$(27)"[30;40m________________________________________________"
30230 PRINT CHR$(27)"["C$(0)"m"CHR$(27)"[24;1H"CHR$(27)"[K     copyright 1986 ABS Logic, Inc."
30240 PRINT CHR$(27)"[25;1H"CHR$(27)"[K parts copyrighted 1982 Microsoft Corp."CHR$(27)"[H"
30250 T=5000:WHILE T>0:T=T-1-T*LEN(INKEY$):WEND:RETURN
```

Print $

Locate 24,1

Foreground

30 - Blk
31 - Red
32 - Green
33 - Yellow
34 - Blue
35 - Magenta
36 - Cyan
37 - White
40 - Background

Color

0 - Black
1 - Blue
2 - Green
3 - Cyan
4 - Red
5 - Magenta
6 - Brown
7 - White

Add 8 for Hi intensity
+16 for Blink

1 - Bold (Hi Intensity

7 - Reverse Video on

0 - Normal

Color foreground, background

Index$ 1 234 567 8
Fix ID REC chars(25#)

NOTES ON THE AIRNAVIGATION PROGRAM

FX$: Fix identifier

"0" - Navaid
"1" - Airport
"2" - Waypoint
"3" - Intersection
"4" - Terminal Waypoint
"5" - End of flight

FLT$: Flight Route

Fix Information:

Min=6 Bytes
Max=24 Bytes

Byte 1 - FX$
Byte 2 - ID 0=Navaid, 1=Airport
Bytes 3 to 5 - 3 letter identifier
Byte 6 - "0" in file, "1" in FLT$
Bytes +1 to +18 - if coord in FLT$

Intersection Info:

Min=20 Bytes
Max=38 Bytes

Byte 1 - FX$
Bytes 2 to 6 - Intersection Name
Byte 7 - Reference Navaid ID
Bytes 8 to 10 - Navaid identifier
Byte 11 - same as Byte 6 above
Bytes +1 to +18 - if coord in FLT$
Bytes 12 to 16 - Radial
Bytes 17 to 20 - DME

Coordinate Information:
(if in FLT$)

Bytes +1 to +7 - Latitude
Bytes +8 to +15 - Longtitude
Bytes +16 to +18 - Variation

FLAGS:

FLG - Set to 1 if Data has been added to the Nav File.
Set to 1 if Coord in FLT$
FLG1 - Set to 1 if Data is to be stored
Set to 1 to return from distance subrutine
FLG2 - Used in flight display routine
FLG3 - Flags departure fix is being entered
FLG4 - Flags second enroute fix has been entered
FLG5 - Return from line 6510 back to route entry

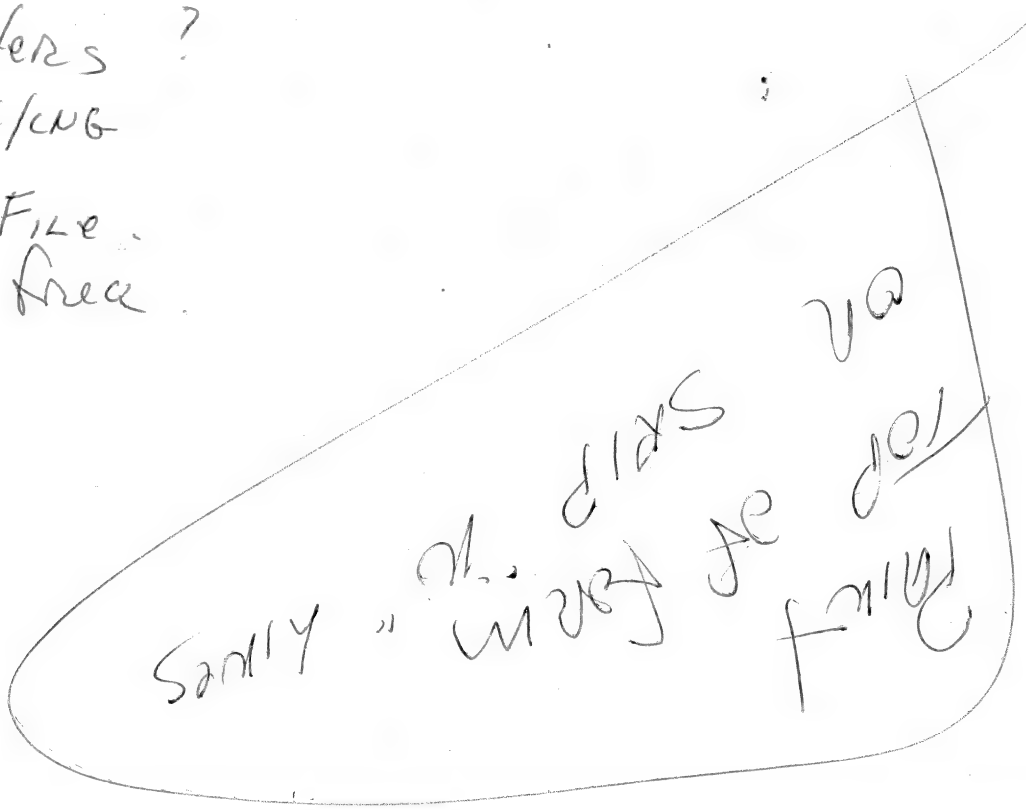
VARIABLES:

P - Printer flag
I - Number of legs
S - Total Distance
WD - Waypoint Leg Distance
WL - Total distance accumulated to waypoints
LL - Leg Length
LD - Leg Distance
A - FLT$ Pointer
D - Line to print
C - Waypoint Counter
RD - Route Direction
FLP$ - Offset list for legs in FLT$

E = Edit
E1 = Edited

Improvements

SORT - # letters ?
negative - Int/CNG
more in File.
Add Redo free.



/* GCIRCLE.C 24-May-86

Please address any bugs, suggestions, or comments to me at:

Paul Mennen
SIGnology, Inc.
1750 Silacci Dr.
Campbell, CA 95008

OR

Paul Mennen
756 Clara Dr.
Palo Alto, CA 94303

Calculates the great circle route between two points P1 and P2
on the surface of a sphere. A great circle route is the shortest
path between two points on the surface of a sphere.

This program is intended as a flight planning tool, but may also
be of use to marine navigators, seismologists, or others interested
in spherical geometry.

You will be prompted for P1 and P2 (the departure point and the
destination point) as well as names for P1 and P2 to make the output
record more meaningful. P1 and P2 are specified by longitude and
latitude in degrees, minutes, and seconds entered in that order.
The degrees, minutes, and seconds should be entered separated by one
or more spaces or tabs, and may be entered with or without a decimal
point. The seconds need not be entered if its zero. If both the minutes
and seconds are zero, then they both may be omitted. For example 67.21
degrees may be entered as:

```
      67.21  0     0
 or   67.21                (the seconds and minutes may be omitted)
 or   67     12.6          (because 12.6 minutes = .21 degrees)
 or   67     12    36      (because 36 seconds = .6 minutes)
```

TO ENTER LATITUDES:
The entry shown above will be interpreted as 67.21 degrees North latitude.
67.21 degrees South latitude is entered as follows:

```
            -67.21
or          -67     12.6
or          -67     12    36
```

TO ENTER LONGITUDES:
The first form shown (positive numbers), is used to enter West longitudes.
The second form (negative numbers), is used to enter East longitudes.

The above convention was chosen to make the program easy to use in the
U.S., since it makes all latitudes and longitudes positive. However,
you may change this convention easily as follows:

1.) If you want to specify North latitudes as negative and South latitudes
as positive then change the first line of this c program from
"double north = 1.;" to "double north = -1.;"
This will be desirable if you live in the southern latitudes.

2.) If you want to specify West longitudes as negative and East longitudes
as positive then change the second line of this c program from
"double east = -1.;" to "double east = 1.;"
This will be desirable if you live in the eastern longitudes.

The output includes a list of points which lie on the great circle. The points listed may be chosen by three different methods:

1.) EQUAL INCREMENTS IN LONGITUDE. With this method, you are asked for a longitude increment. If for example, you specify 1 degree, then the list will include points that are separated by 1 degree of longitude. The points will also be chosen so that they fall on even 1 degree longitude lines. Since the points will fall on the longitude lines of an aeronautical chart, this method will provide an easy way to plot the great circle route on a chart. Note: you must enter the longitude increment in degrees and minutes, where both numbers are integers. If the minutes entry is zero, it may be omitted.

2.) EQUAL INCREMENTS IN LATITUDE. This is similar to the previous method except that you will be prompted for a latitude limit, and the points listed with fall on the latitude lines of your chart. This method will be more useful than method 1 for routes which are nearly north/south. As before, the latitude increments are entered in degrees and minutes, where both numbers are integers.

3.) EQUAL INCREMENTS IN DISTANCE. With this method, you are asked for a distance increment. If for example, you specify 27.3 nautical miles, the points listed will be 27.3 nautical miles apart.

For all three methods, each point listed will show the latitude, longitude, true course, and distance from the departure and destination.

You may skip any of the three methods by entering a carriage return in response to the resolution request (longitude resolution, latitude resolution, distance resolution for methods 1,2 and 3 respectively).

You will also be prompted for an output file for the results, which will also appear on the screen exactly as they will appear in the file. If you do not want to save the results in a file, just respond to the prompt with a carriage return.

Since the derivations of the equations used in the program proved to be interesting, I am including them below. I was also surprised that I could not find these equations published anywhere.

---------- CALCULATE DISTANCE BETWEEN TWO POINTS ON A SPHERE ----------

Let P1 = (X1,Y1,Z1) = (R,G1,T1) [Departure point]
P2 = (X2,Y2,Z2) = (R,G2,T2) [Destination point]

where X,Y,Z are standard rectangular coordinates
R,G,T are spherical coordinates (Radius, lonGitude, laTitude)

Note that the spherical coordinates used here (R,G,T) correspond to (Rho, Theta, 90-Phi) where Rho, Theta, and Phi are the spherical coordinates in common use.

Lets assume a unit sphere (i.e. R=1). Then we have:

X1	=	COS(G1) COS(T1)	X2	=	COS(G2) COS(T2)
Y1	=	SIN(G1) COS(T1)	Y2	=	SIN(G2) COS(T2)
Z1	=	SIN(T1)	Z2	=	SIN(T2)

Since we are on a unit sphere, the distance between P1 and P2 is simply the angle between vectors P1 and P2 (THETA).

P1 . P2 = mag(P1) mag(P2) COS(THETA) = COS(THETA)

(the . means dot product)

or: THETA = DISTANCE = ARCCOS(P1 . P2)

To get the distance on the surface of the earth, we must multiply by the earth's radius (EARTHR).

thus: DISTANCE = EARTHR ARCCOS(X1 X2 + Y1 Y2 + Z1 Z2)

-------- METHOD 1: CALCULATE LATITUDE AS A FUNCTION OF LONGITUDE --------

The equation of the plane formed by P1, P2, and 0 is: (dot product = ".")

N' . X = 0 where N' = P1 x P2 or N . X = 0 where N = N'/mag(N')

Let N' = A'i + B'j + C'k and N = Ai + Bj + Ck (N is a unit vector)

$$\text{Then: } N' = P1 \times P2 = \det \begin{matrix} i & j & k \\ X1 & Y1 & Z1 \\ X2 & Y2 & Z2 \end{matrix}$$

and:
$$A' = Y1\,Z2 - Z1\,Y2 \qquad A = A'/L$$
$$B' = Z1\,X2 - X1\,Z2 \qquad B = B'/L$$
$$C' = X1\,Y2 - Y1\,X2 \qquad C = C'/L$$

where L = SQRT(A'**2 + B'**2 + C'**2)

The equation of the plane in rectangular coordinates is AX + BY + CZ = 0, which gives us the equation of the plane in spherical coordinates as:

A COS(G) COS(T) + B SIN(G) COS(T) + C SIN(T) = 0

or: (A COS(G) + B SIN(G)) COS(T) = -C SIN(T)

or: T = ARCTAN ((A COS(G) + B SIN(G)) / -C)

Which gives the latitude of the great circle route as a function of the longitude.

-------- METHOD 2: CALCULATE LONGITUDE AS A FUNCTION OF LATITUDE --------

Unexpectedly, this calculation turns out to be more tricky than the previous problem. Note that the equation for latitude as a function of longitude can not be solved for longitude as a function of latitude in closed form. Although we could solve for longitude iteratively, using that equation, a different approach will yield a closed form solution.

To appreciate a more subtle difficulty, note that a great circle route that begins by moving towards the east will still be moving towards the east at the end of the route, however a great circle route may begin with the latitude increasing (i.e moving north), and end with the latitude decreasing (i.e. moving south). This makes method 2 more complicated to program.

We are at point P with a known latitude(T). Therefore we know the z coordinate of point P from above:

```
Z = sin(T)              (note that we are still assuming a unit sphere)
```

Since P is on the great circle, it is also on the plane computed above:

```
AX + BY + CZ = 0
```

```
from which we get:   (BY)**2 = (AX + CZ)**2
               or:   (BY)**2 = (AX)**2 + 2ACZX + (CZ)**2
```

Since P is also on the surface of a unit sphere, we have:

```
X**2 + Y**2 + Z**2 - 1 = 0
```

if we multiply this equation by B**2 we get:

```
(BX)**2 + (BY)**2 + (BZ)**2 - B**2 = 0
```

then substituting the previous expression for (BY)**2 into this we get:

```
(A**2 + B**2) X**2  + (2ACZ) X + (C**2 Z**2 + B**2 Z**2 - B**2) = 0
```

which is a quadratic equation for X which can easily be solved. Unfortunately this can result in two real solutions (or perhaps no real solutions). This should have been expected since we have solved for the intersection between two planes and the surface of a sphere. If there are no real roots to this equation, then the specified latitude does not lie on the great circle, and we should take this as a cue to turn around, i.e. if we have been increasing the latitude along our route, we should now start decreasing it. We still have the problem of choosing between the two real roots when they both exist. (If your curious about how this can be done, see the code).

If we call the correct solution to the quadratic equation X, then we can solve for Y with:

```
Y = (AX + CZ) / -B
```

Then the Longitude which we were trying to calculate is simply:

```
G = ARCTAN(Y/X)
```

--- METHOD 3: CALCULATE LATITUDE AND LONGITUDE AS A FUNCTION OF DISTANCE ---

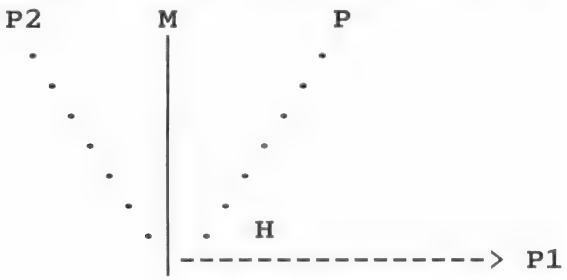
We must find the point P on the great circle route, such that
the distance between P and P1 is a specified distance (DST).
Thus the angle (H) between the two vectors P and P1 is:

$$H = DST/EARTHR.$$

Let $M = N \times P1 = Di + Ej + Fk$ (also a unit vector)

Thus $D = B\,Z1 - C\,Y1$
$E = C\,X1 - A\,Z1$
$F = A\,Y1 - B\,X1$

Let R' be a new coordinate system with unit vectors (i',j',k') = (P1,M,N)



Note that vectors P,P1,P2,M all lie in the same plane.

Thus the coordinates of P in the new coordinate system (R') is:

$$P = \cos(H)\,i' + \sin(H)\,j' = (i'\ j'\ k') \begin{pmatrix} \cos(H) \\ \sin(H) \\ 0 \end{pmatrix} = (P1\ M\ N) \begin{pmatrix} \cos(H) \\ \sin(H) \\ 0 \end{pmatrix}$$

$$= \begin{pmatrix} X1 & D & A \\ Y1 & E & B \\ Z1 & F & C \end{pmatrix} \begin{pmatrix} \cos(H) \\ \sin(H) \\ 0 \end{pmatrix} = \begin{pmatrix} X1\cos(H) + D\sin(H) \\ Y1\cos(H) + E\sin(H) \\ Z1\cos(H) + F\sin(H) \end{pmatrix}$$

The final vector is the location of P in the original coordinate system.
(The matrix (P1 M N) rotates the coordinates from R' to R)

---- CALCULATION OF TRUE COURSE AS A FUNCTION OF POSITION ON THE ROUTE ----

This part of the problem is geometrically the most difficult to visualize.

Let P = current position = (Xi + Yj + Zk)

Since the great circle route lies in the plane N . X = 0, the tangent to the circle at point P (i.e. the direction of travel) also lies in this plane and therefore is perpendicular to N. By definition, the tangent must also be perpendicular to P, thus:

$$\text{Tangent Vector} = T = N \times P = \det \begin{matrix} i & j & k \\ A & B & C \\ X & Y & Z \end{matrix} = Qi + Rj + Sk$$

where: Q = BZ - CY
R = CX - AZ
S = AY - BX

Note: T is a unit vector, since N and P are orthogonal unit vectors

Let V = vertical vector (i.e. north-south axis) = (0,0,1)

Then let W = P x V / mag(P x V). This is a unit vector which is perpendicular to the plane containing P and the north-south axis.

$$P \times V = \det \begin{matrix} i & j & k \\ X & Y & Z \\ 0 & 0 & 1 \end{matrix} \quad \text{thus } W = Yi - Xj \;/\; SQRT(X^{**}2 + Y^{**}2)$$

$$\text{let F = north pointing vector} = W \times P = 1/SQRT(X^{**}2 + Y^{**}2) \det \begin{matrix} i & j & k \\ Y & -X & 0 \\ X & Y & Z \end{matrix}$$

thus F = (- XZi - YZj + (X**2 + Y**2)k) / SQRT(X**2 + Y**2)

Note that F is a unit vector since W and P are orthogonal unit vectors.

Since the true course is the angle between the Tangent vector and the North pointing vector, we have:

T . F = mag(T) mag(F) COS(course) = COS(course)

and finally:

$$\text{course} = \text{ARCCOS}\,(T \cdot F) = \text{ARCCOS}\left(\frac{-QXZ - RYZ + S(X^{**}2 + Y^{**}2)}{SQRT(X^{**}2 + Y^{**}2)} \right)$$

*/

```
double north = 1.;      /* change this to -1 if you live in South Latitudes */
double east = -1.;      /* change this to +1 if you live in East Longitudes */

#include <stdio.h>

#define EARTHR 3440.235     /* radius of earth in Nautical Miles */
#define RADDEG 57.2957795   /* number of degrees in a radian */
#define DEGRAD .017453293   /* number of radians in a degree */
#define RADMIN 3437.74677   /* number of minutes in a radian */
#define MINRAD 2.9088821E-4 /* number of radians in a minute */
#define PI 3.141592654      /* pi   */
#define PI2 6.283185308     /* 2 pi */
#define PID2 1.57079633     /* pi/2 */
#define BEEP 7              /* bell character (control-G)    */

short tj,td,gj,gd,stepd,stepm,m,k;
double t1,td1,tm1,ts1,g1,gd1,gm1,gs1;
double t2,td2,tm2,ts2,g2,gd2,gm2,gs2;
double x1,y1,z1,x2,y2,z2,x,y,z,a,b,c,d,e,f,t;
double aa,bb,cc,xx,yy,gg,b2,oldg;
double g,tm,gm,dst,totald,stepnm,ang,sang,cang,crs,drem;
char chr,file[40],pb[40];
char name1[40] = {'N','o','r','t','h',0,'S','o','u','t','h',0};
char name2[40] = {'E','a','s','t',0,'W','e','s','t',0};
char ns[2] = {'N','S'};
char ew[2] = {'E','W'};
char fmt1[] = "\n%7.1f %7.1f    %c%3d %8.2f      %c%3d %8.2f %10.2f";
char fmt2[] = "%7.2f %5.2f %5.2f  %s\n";
char fmt3[] = "%7.2f %5.2f %5.2f\n";
char fmt4[] = "\n                  ***** GREAT CIRCLE ROUTE *****\n\nDeparture\
   latitude (Deg Min Sec) = ";
char fmt5[] = "           longitude (Deg Min Sec) = ";
char fmt6[] = "Destination latitude (Deg Min Sec) = ";
char fmt7[] = "           longitude (Deg Min Sec) = ";
char fmt8[] = "\n\n\nLongitude resolution = %4d deg, %4d min\n";
char fmt12[] = "\n\n\nLatitude resolution = %4d deg, %4d min\n";
char fmt9[] = "\n    NM     NM        LATITUDE          LONGITUDE\
        TRUE\n   GONE   TO GO   degrees minutes\
    degrees minutes    COURSE\n";
char fmt10[] = "\n\n\nDistance resolution = %3.1f Nautical Miles\n";
char fmt11[] = "\n TOTAL TRIP DISTANCE = %6.3f Nautical miles";

FILE *fp,*fopen();
double sin(),cos(),atan(),sqrt();  /* standard library functions */
double course(),dist(),adif(),rad(),asin(),acos(),atan3(),atan4();



main()
{

printf("\n                    GREAT CIRCLE COMPUTATIONS\n\
             Paul Mennen, SIGnology, Campbell CA.\n\n\
          For Help, see first two pages of gcircle.c\n\n");

printf("  Use negative entries for %s latitudes and %s longitudes\n\n",
         &name1[6*(north > 0.)],&name2[5*(east > 0.)]);
```

```
for (;;)                                                    /* repeat forever  */
{
printf("\nEnter departure point name: ");  gets(name1);  /* get departure   */
printf("Enter departure point latitude (deg min sec): ");
getdms(&td1,&tm1,&ts1,1);  t1 = north * rad(td1,tm1,ts1);
printf("Enter departure point longitude (deg min sec): ");
getdms(&gd1,&gm1,&gs1,0);  g1 = east * rad(gd1,gm1,gs1);

printf("Enter destination point name: ");  gets(name2);  /* get destination */
printf("Enter destination point latitude (deg min sec): ");
getdms(&td2,&tm2,&ts2,1);  t2 = north * rad(td2,tm2,ts2);
printf("Enter destination point longitude (deg min sec): ");
getdms(&gd2,&gm2,&gs2,0);  g2 = east * rad(gd2,gm2,gs2);

stor(&x1,&y1,&z1,g1,t1);                        /* convert to rectangular coord */
stor(&x2,&y2,&z2,g2,t2);                        /* convert to rectangular coord */
printf(fmt11,totald=dist(x1,y1,z1,x2,y2,z2));        /* total trip distance */

for (;;)                                                    /* repeat forever */

{  printf("\nEnter:\n   <CR> = compute great circle\n\
     R  = reenter departure & destination\n\
     Q  = Quit\n      ---> ");

   gets(pb);  chr = pb[0] | 32;     /* get character, convert to lower case */
   if (chr == 'q') return;
   if (chr == 'r') break;
   if (chr == 32)
                                  /* do this if carriage return was entered */
   {  do { printf("\n(CR to skip) Output file: "); gets(file); }
      while (file[0] && !(fp = fopen(file,"w")));
      printf(fmt4);  printf(fmt2,td1,tm1,ts1,name1);
      printf(fmt5);  printf(fmt3,gd1,gm1,gs1);
      printf(fmt6);  printf(fmt2,td2,tm2,ts2,name2);
      printf(fmt7);  printf(fmt3,gd2,gm2,gs2);
      if (fp)
      {  fprintf(fp,fmt4);  fprintf(fp,fmt2,td1,tm1,ts1,name1);
         fprintf(fp,fmt5);  fprintf(fp,fmt3,gd1,gm1,gs1);
         fprintf(fp,fmt6);  fprintf(fp,fmt2,td2,tm2,ts2,name2);
         fprintf(fp,fmt7);  fprintf(fp,fmt3,gd2,gm2,gs2);
         fprintf(fp,fmt11,totald);
      }

      a  =  y1 * z2  -  z1 * y2;        /* compute N' */
      b  =  z1 * x2  -  x1 * z2;
      c  =  x1 * y2  -  y1 * x2;

      dst = sqrt(a*a + b*b + c*c);      /* magnitude of N' */

      a = a/dst; b = b/dst; c = c/dst; /* compute unit vector N */
```

```
/* ----------- METHOD 1 : EQUAL LONGITUDE STEPS -------------------   */

printf("\n(CR to skip) Enter longitude resolution (deg min): ");
gets(pb);  stepd = stepm = 0;  sscanf(pb,"%d%d",&stepd,&stepm);

if (stepd+stepm)       /* skip this section if user entered <cr> or 0 */

{ printf(fmt8,stepd,stepm);  printf(fmt9);
  if (fp) { fprintf(fp,fmt8,stepd,stepm);  fprintf(fp,fmt9); }
  stepm += 60 * stepd;                    /* longitude step in minutes   */
  m = g1 * RADMIN;                        /* initial longitude (minutes) */

  k = ((g=g2-g1) > 0.) ? ((g<PI) ? 1 : -1) : ((g>-PI) ? -1 : 1);
                    /* k = 1 if longitude is increasing, -1 otherwise */

  while (m % stepm) m += k;  /* initial longitude is multiple of stepm */
  if (k == -1) stepm = -stepm;
  g = MINRAD * m; /* convert to radians */

  while (adif(g,g1) < adif(g1,g2)) /* while longitude is within range */

  {  t = atan4((a*cos(g) + b*sin(g)),-c);          /* compute latitude */
     dm(g,&gj,&gd,&gm); dm(t,&tj,&td,&tm);      /* convert to deg,min */
     stor(&x,&y,&z,g,t);              /* convert to rectangular coord.   */
     dst = dist(x1,y1,z1,x,y,z);      /* compute distance from P1        */
     lpoint();                        /* list the point                  */
     g = MINRAD * (m += stepm);       /* advance to next longitude value */
  } /* end of: while (adif(g,g1) > adif(g1,g2))  */

}   /* end of: if (stepd+stepm) */
```

```
/* ------------ METHOD 2 : EQUAL LATITUDE STEPS -------------------   */

printf("\n\n(CR to skip) Enter latitude resolution (deg min): ");
gets(pb);  stepd = stepm = 0;  sscanf(pb,"%d%d",&stepd,&stepm);

if (stepd+stepm)       /* skip this section if user entered <cr> or 0 */

{ printf(fmt12,stepd,stepm);  printf(fmt9);
  if (fp) { fprintf(fp,fmt12,stepd,stepm);  fprintf(fp,fmt9); }
  stepm += 60 * stepd;                        /* latitude step in minutes */
  m = t1 * RADMIN;                          /* initial latitude (minutes) */

  crs = course(x1,y1,z1,a,b,c);  /* is initial course north or south? */
  k = ((crs > 90.) && (crs < 270.)) ? -1 : 1;
       /* k = 1 if latitude is increasing, -1 otherwise */

  while (m % stepm) m += k;  /* initial latitude is multiple of stepm */
  if (k == -1) stepm = -stepm;
  t = MINRAD * m;             /* convert to radians */
  oldg = g = g1;

  while (1)
  {  z = sin(t);   b2 = b*b;
     aa = a*a + b2;  bb = 2. * a * c * z;  cc = z*z*(c*c + b2) - b2;
     xx = bb*bb - 4.*aa*cc;
     if (xx > 0.)                  /* here if quadratic has real roots */
     {  xx = sqrt(xx);
        x = (-bb + xx) / (2.*aa);   xx = (-bb - xx) / (2.*aa);
        y = (a*x + c*z) / -b;       yy = (a*xx + c*z) / -b;
        g = atan3(y,x);             gg = atan3(yy,xx);
        if (adif(gg,oldg) < adif(g,oldg))
        { x = xx;  y = yy;  xx = g;  g = gg;  gg = xx; }
     }
     else       /* here if quadratic has complex roots (turn around!) */
     { g = gg;  t = MINRAD * (m += (stepm = -stepm));
       stor(&x,&y,&z,g,t);           /* convert to rectangular coord. */
     }
     if (adif(g,g1) > adif(g1,g2)) break;
     dm(oldg=g,&gj,&gd,&gm);         /* convert to deg,min */
     dm(t,&tj,&td,&tm);              /* convert to deg,min */
     dst = dist(x1,y1,z1,x,y,z);     /* compute distance from P1 */
     lpoint();                       /* list the point */
     t = MINRAD * (m += stepm);      /* advance latitude */
  } /* end of: while (1)  */

}   /* end of: if (stepd+stepm) */
```

```
   /* ------------ METHOD 3 : EQUAL DISTANCE STEPS -------------------   */

   printf("\n\n(CR to skip) Enter distance resolution in naut. miles: ");
   gets(pb);  stepnm = 0.;  sscanf(pb,"%lf",&stepnm);

   if (stepnm != 0.)      /* skip this section if user entered <cr> or 0 */
   { fprintf(fp,fmt10,stepnm);  fprintf(fp,fmt9);

     d  =  b * z1  -  c * y1;          /* compute unit vector M = N x P1 */
     e  =  c * x1  -  a * z1;
     f  =  a * y1  -  b * x1;

     dst = 0.;  /* initialize distance from P1 */
     while (1)
     {  sang = sin(ang = dst/EARTHR);  cang = cos(ang);

        x  =  x1 * cang  +  d * sang;        /* compute coordinates of P */
        y  =  y1 * cang  +  e * sang;
        z  =  z1 * cang  +  f * sang;

        rtos(&g,&t,x,y,z);           /* convert to spherical coordinates */
        dm(g,&gj,&gd,&gm);           /* convert to degrees,minutes       */
        dm(t,&tj,&td,&tm);           /* convert to degrees,minutes       */
        lpoint();                    /* list the point                   */
        if (dst == totald) break;
        if ((dst += stepnm) > totald) dst=totald;  /* do last point (P2) */
     }  /* end of while (1)        */

   }    /* end of if (stepnm != 0) */

fclose(fp);
}  /* end of if (chr == 32) */

}    /* end of for(;;)         */
}    /* end of for(;;)         */
}    /* end of main            */
```

```
lpoint()     /* list a point on the great circle route */
{  drem = totald-dst;  crs = course(x,y,z,a,b,c);
   printf(fmt1,dst,drem,ns[tj],td,tm,ew[gj],gd,gm,crs);
   if (fp) fprintf(fp,fmt1,dst,drem,ns[tj],td,tm,ew[gj],gd,gm,crs);
}


double course(x,y,z,a,b,c)
                  /* Returns the true course at point (x,y,z) of the great */
                  /* circle route defined by the plane ax + by + cz = 0    */
double x,y,z,a,b,c;
{  double q,r,s,tmp1,tmp2;
   q  =  b * z  -  c * y;             /* compute T = N x P (tangent vector) */
   r  =  c * x  -  a * z;
   s  =  a * y  -  b * x;
   tmp2 = sqrt(tmp1 = x*x + y*y);     /* compute true course */
   tmp1 = RADDEG * acos((s*tmp1 - q*x*z - r*y*z)/tmp2);
   return((c > 0) ? tmp1 : 360. - tmp1);
}


getdms(d,m,s,lat)     /* get degrees,min,seconds from one line of user input */
double *d,*m,*s;
short lat;          /* true if entry is for a latitude, false for longitude */
{ short err;
  err = 0;
  do
  { if (err) printf("%c                              Bad entry, try again: ",BEEP);
    gets(pb);  *d = *m = *s = 0.;  sscanf(pb,"%lf%lf%lf",d,m,s);  err = 1;
  }
  while ((*m > 60.) || (*m < 0.) || (*s > 60.) || (*s < 0.) ||
         (*d > 180.) || (*d < -180.) || (lat && ((*d > 90.) || (*d < -90.))));
}


double dist(x,y,z,x2,y2,z2)            /* compute distance between P and P2 */
double x,y,z,x2,y2,z2;
{ return(EARTHR * acos(x*x2 + y*y2 + z*z2)); }


stor(x,y,z,g,t)                         /* convert spherical to rectangular */
double *x,*y,*z,g,t;
{  double temp;
   *x = cos(g) * (temp = cos(t));
   *y = sin(g) * temp;
   *z = sin(t);
}

rtos(g,t,x,y,z)                         /* convert rectangular to spherical */
double *g,*t,x,y,z;
{  *g=atan3(y,x);
   *t=asin(z);
}
```

```
double rad(d,m,s)                          /* convert deg,min,sec to radians */
double d,m,s;
{  m = m/60. + s/3600.;  return(DEGRAD * ((d>0.) ? (d+m) : (d-m))); }


dm(r,sign,deg,min) /* convert angle r (in radians) to degrees and minutes   */
                   /* sign is returned as 0 if r>=0 and as 1 if r<0         */
                   /* nonnegative numbers are always returned for deg & min */
double r,*min;
short *sign,*deg;
{  *sign = r < 0.;
   r *= (r>0.) ? RADDEG : -RADDEG;
   *min = 60. * (r - (*deg = r));
   if (*min > 59.995) { *deg += 1; *min = 0.; }
}


double adif(a1,a2)                  /* Returns difference between two angles */
double a1,a2;
{  double a;
   a = abs(a1-a2);
   while (a > PI) a = abs(a - PI2);
   return(a);
}


double asin(x)
double x;
{ double a; if ((a = 1. - x*x) < 0.0) a = 0.; return(atan3(x,sqrt(a))); }

double acos(x) double x;  { return(PID2 - asin(x)); }

double atan3(y,x)                    /* returns and angle between -pi and pi */
double y,x;                          /* note: this is usually called atan2   */
{ return( (x >= 0.)
    ? ((x <  1.e-30) ?  PID2 : atan(y/x))
    : ((x > -1.e-30) ? -PID2 : atan(y/x) + ((y>=0.) ? PI : -PI)));
}

double atan4(y,x)                 /* returns and angle between -pi/2 and pi/2 */
double y,x;                       /* which is the same range returned by atan */
{ return( (x >= 0.)
    ? ((x <  1.e-30) ?  PID2 : atan(y/x))
    : ((x > -1.e-30) ? -PID2 : atan(y/x)));
}
```

/* GCIRCLE.C 24-May-86

Please address any bugs, suggestions, or comments to me at:

Paul Mennen
SIGnology, Inc.
1750 Silacci Dr.
Campbell, CA 95008

OR

Paul Mennen
756 Clara Dr.
Palo Alto, CA 94303

Calculates the great circle route between two points P1 and P2
on the surface of a sphere. A great circle route is the shortest
path between two points on the surface of a sphere.

This program is intended as a flight planning tool, but may also
be of use to marine navigators, seismologists, or others interested
in spherical geometry.

You will be prompted for P1 and P2 (the departure point and the
destination point) as well as names for P1 and P2 to make the output
record more meaningful. P1 and P2 are specified by longitude and
latitude in degrees, minutes, and seconds entered in that order.
The degrees, minutes, and seconds should be entered separated by one
or more spaces or tabs, and may be entered with or without a decimal
point. The seconds need not be entered if its zero. If both the minutes
and seconds are zero, then they both may be omitted. For example 67.21
degrees may be entered as:

```
      67.21  0     0
 or   67.21                (the seconds and minutes may be omitted)
 or   67     12.6          (because 12.6 minutes = .21 degrees)
 or   67     12    36      (because 36 seconds = .6 minutes)
```

TO ENTER LATITUDES:
   The entry shown above will be interpreted as 67.21 degrees North latitude.
   67.21 degrees South latitude is entered as follows:

```
            -67.21
   or       -67     12.6
   or       -67     12    36
```

TO ENTER LONGITUDES:
   The first form shown (positive numbers), is used to enter West longitudes.
   The second form (negative numbers), is used to enter East longitudes.

The above convention was chosen to make the program easy to use in the
U.S., since it makes all latitudes and longitudes positive. However,
you may change this convention easily as follows:

1.) If you want to specify North latitudes as negative and South latitudes
    as positive then change the first line of this c program from
        "double north = 1.;" to "double north = -1.;"
    This will be desirable if you live in the southern latitudes.

2.) If you want to specify West longitudes as negative and East longitudes
    as positive then change the second line of this c program from
        "double east = -1.;" to "double east = 1.;"
    This will be desirable if you live in the eastern longitudes.

The output includes a list of points which lie on the great circle. The points listed may be chosen by three different methods:

1.) EQUAL INCREMENTS IN LONGITUDE. With this method, you are asked for a longitude increment. If for example, you specify 1 degree, then the list will include points that are separated by 1 degree of longitude. The points will also be chosen so that they fall on even 1 degree longitude lines. Since the points will fall on the longitude lines of an aeronautical chart, this method will provide an easy way to plot the great circle route on a chart. Note: you must enter the longitude increment in degrees and minutes, where both numbers are integers. If the minutes entry is zero, it may be omitted.

2.) EQUAL INCREMENTS IN LATITUDE. This is similar to the previous method except that you will be prompted for a latitude limit, and the points listed with fall on the latitude lines of your chart. This method will be more useful than method 1 for routes which are nearly north/south. As before, the latitude increments are entered in degrees and minutes, where both numbers are integers.

3.) EQUAL INCREMENTS IN DISTANCE. With this method, you are asked for a distance increment. If for example, you specify 27.3 nautical miles, the points listed will be 27.3 nautical miles apart.

For all three methods, each point listed will show the latitude, longitude, true course, and distance from the departure and destination.

You may skip any of the three methods by entering a carriage return in response to the resolution request (longitude resolution, latitude resolution, distance resolution for methods 1,2 and 3 respectively).

You will also be prompted for an output file for the results, which will also appear on the screen exactly as they will appear in the file. If you do not want to save the results in a file, just respond to the prompt with a carriage return.

Since the derivations of the equations used in the program proved to be interesting, I am including them below. I was also surprised that I could not find these equations published anywhere.

---------- CALCULATE DISTANCE BETWEEN TWO POINTS ON A SPHERE ----------

Let P1 = (X1,Y1,Z1) = (R,G1,T1) [Departure point]
P2 = (X2,Y2,Z2) = (R,G2,T2) [Destination point]

where X,Y,Z are standard rectangular coordinates
R,G,T are spherical coordinates (Radius, lonGitude, laTitude)

Note that the spherical coordinates used here (R,G,T) correspond to (Rho, Theta, 90-Phi) where Rho, Theta, and Phi are the spherical coordinates in common use.

Lets assume a unit sphere (i.e. R=1). Then we have:

X1	=	COS(G1) COS(T1)	X2	=	COS(G2) COS(T2)
Y1	=	SIN(G1) COS(T1)	Y2	=	SIN(G2) COS(T2)
Z1	=	SIN(T1)	Z2	=	SIN(T2)

Since we are on a unit sphere, the distance between P1 and P2 is simply the angle between vectors P1 and P2 (THETA).

P1 . P2 = mag(P1) mag(P2) COS(THETA) = COS(THETA)
(the . means dot product)

or: THETA = DISTANCE = ARCCOS(P1 . P2)

To get the distance on the surface of the earth, we must multiply by the earth's radius (EARTHR).

thus: DISTANCE = EARTHR ARCCOS(X1 X2 + Y1 Y2 + Z1 Z2)

-------- METHOD 1: CALCULATE LATITUDE AS A FUNCTION OF LONGITUDE --------

The equation of the plane formed by P1, P2, and O is: (dot product = ".")

N' . X = O where N' = P1 x P2 or N . X = O where N = N'/mag(N')

Let N' = A'i + B'j + C'k and N = Ai + Bj + Ck (N is a unit vector)

Then: N' = P1 x P2 = det | i j k |
| X1 Y1 Z1 |
| X2 Y2 Z2 |

and: A' = Y1 Z2 - Z1 Y2 A = A'/L
B' = Z1 X2 - X1 Z2 B = B'/L
C' = X1 Y2 - Y1 X2 C = C'/L

where L = SQRT(A'**2 + B'**2 + C'**2)

The equation of the plane in rectangular coordinates is AX + BY + CZ = O, which gives us the equation of the plane in spherical coordinates as:

A COS(G) COS(T) + B SIN(G) COS(T) + C SIN(T) = O

or: (A COS(G) + B SIN(G)) COS(T) = -C SIN(T)

or: T = ARCTAN ((A COS(G) + B SIN(G)) / -C)

Which gives the latitude of the great circle route as a function of the longitude.

-------- METHOD 2: CALCULATE LONGITUDE AS A FUNCTION OF LATITUDE --------

Unexpectedly, this calculation turns out to be more tricky than the previous problem. Note that the equation for latitude as a function of longitude can not be solved for longitude as a function of latitude in closed form. Although we could solve for longitude iteratively, using that equation, a different approach will yield a closed form solution.

To appreciate a more subtle difficulty, note that a great circle route that begins by moving towards the east will still be moving towards the east at the end of the route, however a great circle route may begin with the latitude increasing (i.e moving north), and end with the latitude decreasing (i.e. moving south). This makes method 2 more complicated to program.

We are at point P with a known latitude(T). Therefore we know the z coordinate of point P from above:

```
Z = sin(T)              (note that we are still assuming a unit sphere)
```

Since P is on the great circle, it is also on the plane computed above:

```
AX + BY + CZ = 0
```

```
from which we get:   (BY)**2 = (AX + CZ)**2
               or:   (BY)**2 = (AX)**2 + 2ACZX + (CZ)**2
```

Since P is also on the surface of a unit sphere, we have:

```
X**2 + Y**2 + Z**2 - 1 = 0
```

if we multiply this equation by B**2 we get:

```
(BX)**2 + (BY)**2 + (BZ)**2 - B**2 = 0
```

then substituting the previous expression for (BY)**2 into this we get:

```
(A**2 + B**2) X**2  + (2ACZ) X + (C**2 Z**2 + B**2 Z**2 - B**2) = 0
```

which is a quadratic equation for X which can easily be solved. Unfortunately this can result in two real solutions (or perhaps no real solutions). This should have been expected since we have solved for the intersection between two planes and the surface of a sphere. If there are no real roots to this equation, then the specified latitude does not lie on the great circle, and we should take this as a cue to turn around, i.e. if we have been increasing the latitude along our route, we should now start decreasing it. We still have the problem of choosing between the two real roots when they both exist. (If your curious about how this can be done, see the code).

If we call the correct solution to the quadratic equation X, then we can solve for Y with:

```
Y = (AX + CZ) / -B
```

Then the Longitude which we were trying to calculate is simply:

```
G = ARCTAN(Y/X)
```

--- METHOD 3: CALCULATE LATITUDE AND LONGITUDE AS A FUNCTION OF DISTANCE ---

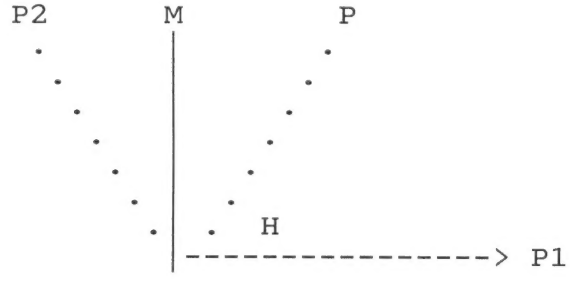
We must find the point P on the great circle route, such that
the distance between P and P1 is a specified distance (DST).
Thus the angle (H) between the two vectors P and P1 is:

$$H = DST/EARTHR.$$

Let $M = N \times P1 = Di + Ej + Fk$ (also a unit vector)

Thus $D = B\,Z1 - C\,Y1$
$E = C\,X1 - A\,Z1$
$F = A\,Y1 - B\,X1$

Let R' be a new coordinate system with unit vectors (i',j',k') = (P1,M,N)



Note that vectors P,P1,P2,M all lie in the same plane.

Thus the coordinates of P in the new coordinate system (R') is:

$$P = \cos(H)\,i' + \sin(H)\,j' = (i'\ j'\ k') \begin{pmatrix} \cos(H) \\ \sin(H) \\ 0 \end{pmatrix} = (P1\ M\ N) \begin{pmatrix} \cos(H) \\ \sin(H) \\ 0 \end{pmatrix}$$

$$= \begin{pmatrix} X1 & D & A \\ Y1 & E & B \\ Z1 & F & C \end{pmatrix} \begin{pmatrix} \cos(H) \\ \sin(H) \\ 0 \end{pmatrix} = \begin{pmatrix} X1\cos(H) + D\sin(H) \\ Y1\cos(H) + E\sin(H) \\ Z1\cos(H) + F\sin(H) \end{pmatrix}$$

The final vector is the location of P in the original coordinate system.
(The matrix (P1 M N) rotates the coordinates from R' to R)

---- CALCULATION OF TRUE COURSE AS A FUNCTION OF POSITION ON THE ROUTE ----

This part of the problem is geometrically the most difficult to visualize.

Let P = current position = (Xi + Yj + Zk)

Since the great circle route lies in the plane N . X = 0, the tangent to the circle at point P (i.e. the direction of travel) also lies in this plane and therefore is perpendicular to N. By definition, the tangent must also be perpendicular to P, thus:

$$\text{Tangent Vector} = T = N \times P = \det \begin{matrix} i & j & k \\ A & B & C \\ X & Y & Z \end{matrix} = Qi + Rj + Sk$$

where: Q = BZ - CY
R = CX - AZ
S = AY - BX

Note: T is a unit vector, since N and P are orthogonal unit vectors

Let V = vertical vector (i.e. north-south axis) = (0,0,1)

Then let W = P x V / mag(P x V). This is a unit vector which is perpendicular to the plane containing P and the north-south axis.

$$P \times V = \det \begin{matrix} i & j & k \\ X & Y & Z \\ 0 & 0 & 1 \end{matrix} \quad \text{thus } W = Yi - Xj \;/\; SQRT(X^{**}2 + Y^{**}2)$$

let F = north pointing vector = W x P = $1/SQRT(X^{**}2 + Y^{**}2)$ $\det \begin{matrix} i & j & k \\ Y & -X & 0 \\ X & Y & Z \end{matrix}$

thus F = $(- XZi - YZj + (X^{**}2 + Y^{**}2)k) \;/\; SQRT(X^{**}2 + Y^{**}2)$

Note that F is a unit vector since W and P are orthogonal unit vectors.

Since the true course is the angle between the Tangent vector and the North pointing vector, we have:

T . F = mag(T) mag(F) COS(course) = COS(course)

and finally:

$$\text{course} = \text{ARCCOS}\,(T \cdot F) = \text{ARCCOS}\left(\frac{-QXZ - RYZ + S(X^{**}2 + Y^{**}2)}{SQRT(X^{**}2 + Y^{**}2)} \right)$$